INTEREX

LAS VEGAS

Computing Management Symposium

March 12-14 1990

Proceedings

Sponsored by INTEREX

The International Association of Hewlett-Packard Computer Users

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

INTEREX

the International Association of
Hewlett-Packard Computer Users

Proceedings

of the

1990 INTEREX
Computing Management Symposium

at

Las Vegas, Nevada
March 12-14, 1990

INDEX BY PAPER NUMBER

INDEX BY PAPER NUMBER

MANAGEMENT SKILLS FOR PERSONAL GROWTH

BUSINESS MANAGEMENT SKILLS

MANAGEMENT INFORMATION SYSTEMS

OFFICE AUTOMATION/PC INTEGRATION

DATA CENTER MANAGEMENT/NETWORKING

MANAGEMENT SOLUTIONS TO BUSINESS PROBLEMS

HP STRATEGIES

Writing for Management Success:
Power is in the Process

Deborah Elwell Arfken, Ed. D.
The University of Tennessee at Chattanooga
615 McCallie Avenue
Chattanooga, Tennessee 37403
615/755-4667

Writing Is Part of Our Work

"Data Center Management and Efficiency." "The Acquisition of Software." "ARPA Services for the HP 3000." " Writing for Management Success." Writing for Management Success? What is writing for management success doing on the program schedule for a computing management symposium?

We should all be interested in this topic for two simple but compelling reasons: 1) an article in The Harvard Business Review stated that the single factor that characterizes a promotable manager is the ability to write well, and 2) ineffective writing is expensive. Let's discuss these facts in more depth.

In an article in a 1982 issue of College English, two professors reported that they had surveyed 200 college-educated working people to find out how much of their time on the job they spent writing. Their findings revealed that people in professional and technical occupations--the types of occupations in which over half of college-trained people are employed--write, on the average, nearly 30 percent of their total work time. Given that college educated people are likely to be in careers which require high abstraction, those careers which are involved in information processing, it is not surprising to learn that their work will require extensive written communication.

In addition to the correlation of kind of work with *amount* of writing performed, research also shows that as one's income rises so does the *frequency* of writing.

And then, secondly, there's always the bottom-line issue of money. According to Toffler, a well-known futurist, the cost of preparing a single business letter ran from $14-18 in 1980 and that was ten years ago. One corporation--a division headquarters--studied its communications. They found that 85% of the letters, memos, and reports written for another's signature were rejected. The signer didn't like the way something was said. These rejected memos and letters had to be revised an average of five times. The cost of each draft equaled $5-10. This division of a well-managed company sacrificed more than $8-16 million annually because of poor writing. Even in a well-managed corporation, it's clear that the cost of ineffective writing is expensive.

The truth of the matter is that our professional image is projected by speaking and writing, which are often more visible than we are. These words, especially these written words, reflect the quality of our thoughts. For too long we have been taught that writing well is solely a matter of *skills*. Let me share another approach with you: writing well is also a matter of *attitude*. And, most of all, it's a matter of *choices*.

The Workshop Organization

This presentation emphasizes a practical approach to mastering the essentials of memo and report writing--the skills and the attitudes. We will focus on the choices in both content and process--what to do and how to do it. The basic layout for the workshop is as follows:

I. Preparing to Write: A Common Sense Approach

- Audience Analysis
- Style/Tone
- Organization
- Prewriting

II. Getting Down to Business: Writing the Draft

III. Reconsidering: a Close Look at the Draft
- The Three C's: Clarity, Conciseness, Completeness
- Design
- Effectiveness

IV. The Polishing Touch: Revising and Editing
- Do's and Don'ts
- Simplicity
- You Attitude
- Transitions

Stage One: Preparing to Write

Let's begin with first things first. Effective writing gets its point across. It communicates with the reader: it moves the reader in the direction the writer desires. To accomplish this goal, the writer must prepare carefully. Dr. Ed. Mullins of the University of Alabama School of Communications said once, "What separates winners from losers is not the will to win, but the will to prepare." If you prepare, no one but you will know how messy your mind was at first.

Ask the right questions. **Who is your audience?** Is it a lay person with no explicit knowledge or interest in your subject? An expert who possesses substantial knowledge, interest and probably experience in the topic? An executive who may not have the knowledge and experience about the topic but who has definite decision-making power over your career? Or is it a technician who has hands-on experience with this topic?

Audience analysis is never simple. While it is possible to determine the primary audience, it is another situation to guess at possible secondary audiences--those who may receive copies from the primary reader or who may obtain the memo or report at a later date from a file somewhere. You may even have a complex audience, composed of people who represent a couple of levels of the audience described above, or you

may have multiple audiences, meaning several people in different categories.

A keen audience analysis has profound implications for determining answers to the next set of questions. **What is the message of this writing?** In other words, *what* do you want to say? **What is the purpose?** *Why* do you want to write this? **What is your role as writer?** What is your *relationship* to this message and purpose? **What style and tone should you use?** What will be the most effective *level of language* to use?

We need to apply all of these questions to the first type of writing that we will discuss in the workshop, the memorandum, better known as the memo. *Memorandum* is a Latin word signifying that something is to be remembered. At we know it today, the memo serves many functions. It is more streamlined and less formal than the report, the other type of writing which we will examine. Although, basically, the memo has a simple format, there are several guidelines which must be followed. An effective memo possesses the following characteristics:

- has a specific subject (RE) line in its heading
- covers only one topic
- puts the conclusion first
- excludes irrelevant details
- is written in an easy, conversational style.
- ends with a call to action that clearly states what the reader is expected to do

Stage Two: Writing the Draft

To begin the process of writing a memo or report, decide what the most important point is and place it first. Continue to write the draft version quickly, even though this advice would bring admonishments from your former high school English teachers. If you dwaddle over your writing, your ideas may vanish. So, for the sake of the flow, write quickly and don't worry about mistakes. There will be time to correct those later.

Stage Three: Reconsidering the Draft

When you reconsider the draft, concentrate on the three c's of good writing: **clarity, conciseness, and completeness**. To achieve clarity, avoid corrupting the language with words such as *career alternative enhancement program*--camouflage verbiage used at Chrysler AMC plants to tell workers that the plant was closing and they were out of jobs. When you mean *capital punishment* be clear and say that--not *our society's recognition of the sanctity of human life.* Transitions, those words like *consequently* and *as a result* serve as connectors, helping to make sentences and paragraphs relate to each other. Be concise: say *now* , not *this point in time.* A useful guide is to write no more than 11-15 words per sentence for maximum readability and conciseness. Finally, be complete: include all necessary information. Have you answered the journalist's questions: who? what? when? where? why? how? And have you used the inverted pyramid order by placing the conclusion first? The power of an effective memo comes from this precise ordering.

Stage Four: The Polishing Touch

Once you've reconsidered the draft, it's time to revise and edit it. The handouts that accompany this workshop point out the do's and don't of using language today. It's also important to emphasize, however, the use of gender neutral language in our writing today. Whenever possible, use sentences like this: *Writers must consider their audience* instead of *a writer must consider his audience.*

The use of simple language is a virtue. How would you like to enter a polling booth and encounter something like this:

> Shall a resolution passed by the Board of Commissioners of Hamilton County on June 20, 1988, numbered 688-54 and published in The Chattanooga Times and Chattanooga News-Free Press, newspapers of general circulation in Hamilton County, which levied an additional tax on the same privileges subject to the Retailers' Sales Tax Act under

> Chapter 6, Parts 1-6, Titles 67, T. C. A., as the same may be amended which are exercised in Hamilton County, to be levied and collected as provided in the Act at an amended rate of nine-twentyseconds of the rate levied in the Retailers' Sales Tax Act, as amended, except as limited or modified by statute, become operative?

This one interrogatory sentence consists of 106 words, 13 commas, 3 dashes, 3 periods, 1 question mark,and a total absence of good writing sense. To keep sentences simple, remember the guideline of 11-15 words per sentence. To keep paragraphs simple, compose them of only 4-5 sentences.

Have you maintained a "you" attitude? When you congratulate your colleague, do you say, *"I want to congratulate you on receiving the contract? "* Or do you say, *"Congratulations on receiving the contract? "* and *"Congratulations! You won the contract! "* Effective writing always considers the reader first, not the writer and his or her ego. A good rule of thumb is to use approximately six *you* words in every 100 words of text.

Let's look at this stage of the writing process even more closely by relating it to report writing. Good report writing follows many of the same guidelines of good memo writing; but, of course, the document--including proposals, feasibility studies, progress reports, meeting reports, and others--is remarkably longer. To this point, the stages of writing a report will have been very similar to those of the memo, although the preparation may have necessitated extensive research. At the revision stage, however, you will need to pay particular attention to the organization of this long document, for the most significant difference of a report is its format. Reports usually follow this standard framework:

I. Front Matter

- Title Page
- Letter of Transmittal

- List of Illustrations
- List of Contents
- Abstract

II. Text

- Introduction
- Body
- Conclusion
- Recommendations

III. Back Matter

- Glossary
- Appendices
- Bibliography

As in much other writing, common sense rules. Always consider your audience, your readers, who are usually several busy executives. The report will be formal in appearance, often in a bound format. As noted above, the first section of the report will feature an executive summary or abstract, a brief overview of the problem and conclusion which highlights the content of the report.

As you might expect, the second section of the text is lengthy since it must explain the rationale for the report, define the problem and situation, provide a review of recent studies, draw conclusions, and make recommendations. Because this section is lengthy, it will be unwieldy for the reader unless you provide plenty of signposts to direct the message. Does the design layout emphasize the content? Does it package your information for easy reading? Specifically, is there plenty of white space? Have stylistic features like bold face and italics been used to emphasize key points? Have lists with bullets and/or numbers been used to control sequential information? Do boxes outline summary information? Above all, have headings and subheadings been used to breakup large amount of content into smaller, more manageable pieces?

The third section, the back matter, includes materials which provide helpful reference. In this section, you can place materials which support the body of the report but which would clutter it up unnecessarily if actually placed there. The hallmark of a good report is that it is *self-contained:* anything that the reader needs to know about the material to understand its background, its implications, and your conclusions is included in the document. Much of the detail of this information will be stored in this section.

A Final Caveat

Nobody ever said that good writing is easy. It's the result of good thinking, which isn't easy either. But we know now that good thinking and good writing are the result of keen audience analysis, a *you* attitude, and sound choices. And we know that good writing substantially influences our careers, for it is the most powerful way to create a positive image for ourselves and our organizations.

Note: Because this session was a workshop, much of the information was delivered on a spontaneous basis to accompany the numerous handouts and practical exercises.

THE TECHNICAL TO MANAGER TRANSITION

An Issue of Changing Roles

Ronald E.Stanton, CDP
REB Software Products

1276 Fairport Road
P.O. Box 66
Fairport, New York 14450-0066
United States of America

INTRODUCTION

Many of today's Data Processing technocrats have developed a strong urge to become managers after limited development experience. Their idealistic views and youthful encounters with the system lead them to a belief that they will be able to change things. After migrating into management, most will find themselves wading in a sea of bewilderment. Just keeping up with day-to-day activities will normally require more than a normal workday. The changes they dreamed of will be put off in lieu of basic survival tactics and a new desire to avoid distasteful encounters with superiors. Most of their available time will seem to be eaten up by meetings and impromptu visits by frustrated users.

As a manager, the role requires a new perspective and the ability to act as a buffer between the troops and the front office. This adjustment can create an enormous amount of stress and foster an identity crisis capable of discouraging most new managers. By viewing the differences between the technical role and a management role, we may be able to identify some key success factors. Keeping these factors in our thoughts as we enter any leadership role may aid us in accomplishing some of the changes we hope to initiate.

UNDERSTANDING THE TECHNICAL ROLE

As technocrats, we normally received concentrated training and grooming that was required in our technical role. The training helped us to focus in on specifics and apply highly specialized tools in our technical solutions. We didn't rely upon a delivery system, we were an integral part of it.

We normally worked within published guidelines and openly shared our concerns with our peers.

User communications normally evolved around technical issues and invariably, we'd find ourselves unable to address their needs. Without management approval, we couldn't officially commit to an enhancement or the correction of program bugs.

From time-to-time, our users also informed us about some deadlines that we weren't aware of. They also were able to let us know about some projects that we never even heard about. These types of conversations were always a delight, especially when we went charging into our manager's office to find out the 'real' story. It's really funny, we never seemed to get a straight answer when we challenged managers about their style.

Project tracking was done by those pesty project management packages that spit out whatever garbage that you put into them. Our managers always asked us to be honest with our input and they seemed to have a fit when we did. They would normally show up at regular intervals and ask us the same mundane questions. The real kicker was the one that related to why it seemed to take such a long time to get anything done. We eventually caught on to the fact that if we said we were on schedule and didn't think we would run into any show-stoppers, they would quietly go away and let us get our jobs done.

Technically, we generally felt very secure and responded well to management's tendency to consistently ask for the impossible. We also gained a wealth of knowledge on how the total organization functioned. The combination of technical skills and functional awareness played a key part in our ability to identify workable solutions that wouldn't require massive re-training and procedural changes.

Even though we often felt that management didn't appreciate the pressures we encountered and the enormous size of our workload, we managed to cope. When our technical tasks were completed and installed, we received our own reward, a sense of self-gratification. It was always fun to think back on all of the barriers and problems that we encountered along the way to success. We never really talked about the short-cuts that were taken or the questionable tactics that were employed along the way. We also felt that our managers just preferred to keep those things quiet; unless we accidentally screwed up the works!

Adaptability seemed to be a major requirement for technical personnel. Our equipment, offices, managers, you name it, were always changing. It was always easier to move the programmers than it was to separate the user groups. As far as all of those promises that were verbalized along the way, it seemed that no one was ever informed when those changes took place. Being on maintenance projects for three years implied that we might get cobwebs before a promotion. Those performance appraisals never seemed to get done on-time either. Policies seemed to be used to punish technicians instead of backing them up. The deadlines for projects were seldom compromised, most other deadlines seemed to come and go without any resolution.

After all of the griping, moaning, and irate discusions, a majority of technochrats still wanted to push for a management position. Whether we played the game or not, we always wanted to be perceived as 'management material' in order to break into the 'big money' brackets. The remainder of this paper is dedicated to those of us who made the transition and were forced to cope with the role changes. It is also dedicated to those who are considering the role transition so that they might gain some insight into one of the most dramatic career adjustments they'll ever encounter.

UNDERSTANDING THE MANAGERMENT ROLE

The dream comes true and a management title is printed on our new business cards. Instead of just two-hundred, we are overwhelmed with a box of a thousand and a nice brass-tone carrying case. We remain in a euphoric state for a few days until management decides that the honeymoon period has to end. We have a hard time arranging the desk-set on the new overhang desk and decide that it's best not to have the PC on it. Within the first week, we were able to make more decisions than we did in the last few months prior to our promotion. We begin to think that this job was intended for us and that our talents will finally be utilized.

Attending our first staff meeting was a blast. Discussions regarding people performance, profits, potential cut-backs, and a perception of low productivity levels seemed to be as vital to the company as our marketing strategies. Several battles broke out and had to be curtailed by the CFO. After leaving the conference room, we wondered why the meeting ran over more than two hours. Didn't they realize that people had other plans during lunch hour? An unstructured meeting was the last thing we expected, and yet, the staff meeting seemed to be very unprofessional.

One of the things that amazed us about the meeting was the cold and almost un-caring environment. Comments were picked out of context and used as battering rams against people. Everyone seemed to go along with that type of discussion. After overhearing a positive comment to a subordinate prior to the meeting, the same manager stated that the employee was marginal and should be considered for a RIF. The realm we just entered surely wasn't based on Hoyle or any other specific set of rules. It was based on survival, without full regard to support personnel.

Our spouse really didn't seem to understand the new position requirements and its' relationship to the organization. We tried to express our views and all we got in return was a comment about just giving it a little more time to work out. After only three weeks in the job we begin to doubt whether we made the right decision. Why did we give up the comfort of all those technical skills and move into a world apart from reality. What could be and what should be seemed to be set aside for what is. We also hear that we have to see things from their perspective and forget what we did prior to becoming a manager. Why doesn't anyone seem to hear what we are trying to say?

After six months on the job, we have asked several times to be enrolled into the management training program. The response concludes that although they agree that it would be beneficial, there wasn't any money in the budget for non-technical seminars. The entire training and travel budgets were being watched very closely and virtually put on-hold. We wonder how they expect a new manager to get the job done without any type of formal training. We finally understand what it means to be 'thrown to the wolves'.

As a manager, we are requested to view our former peers as resources instead of people. We have X resources and we should see X-days worth of output from them every month. We should also continue to ask them for more, even though it is implied that they are expected to work over forty hours a week. Our job is defined to be company-oriented instead of just thinking about the personnel. We need to get as much as we can from them and then ask them for just a little bit more. The technocrat in us wants us to speak out and the management aspirations tell us to keep our mouth shut. We seem to dwell on discipline and negative topics instead of the motivational things that we all read about. What about Japan, do they do that there? Oh, oh, we just reviewed a film on the Japanese worker, they seem to have less free time than the average American. Does management think that our workforce will accept peer pressures to work from fifty to seventy hours a week? Worse yet, what do they think of me when I roll in a few minutes late every day and try to get out just before quitting time?

The technical world had its' rough times, but they were nothing compared to what we encountered in our first year as a manager. Worse yet, our technical capabilities have been slipping away and we will soon be trapped in management without a fall-back position.

Our first review didn't help matters. Why didn't they tell us some of those things six months ago so that we could work on them and get a better review? Why surprise us when they preach about not liking surprises? We begin to question whether there are any rules in the management arena. If there are, why don't they follow them?

Although the previous paragraphs paint a very negative view of a first-line manager's job, it isn't far from reality in many companies. Constant pressure from competitors and the daily cost of operating in an unsure economy have caused many companies to change their attitudes towards employees.

CONCLUSION

Achieving success as a manager may require more adjustments than we ever anticipated. There aren't any clear rules or guidelines to follow and we never seem to know if we did the right thing. The technical role we used to have portrayed us as a doer. Although this does not imply that managers are not doers, it just means that their respective role is interpreted differently.

As a manager, it may be more beneficial to be concerned about how we are perceived. More promotions appear to be earned by a positive image than on other factors. No matter how hard we work or how many hours of extra time we put-in, we may not gain the recognition we deserve. It behoves us to be 'visible' in a positive way. Other managers have to see us in off-hours and we should casually mention the extra week-end time in routine business situations. We have to be concerned about how people feel about us. If we aren't the team-oriented can-do image the organization wants, we may never move up. Mobility is within our grasp, we just have to become a salesperson and promote our own potential.

When a fast-tracker begins moving within the structure, we should educate ourselves by analyzing the key factors that might have contributed to their mobility. We have to be open-minded and avoid some of the petty thoughts about why a person moved up. Responding in a positive way will help to project a positive image and help contribute to our success. A good motto to follow is to be happy for all who taste the flavor of success. It may earn you a promotion someday. Also, you never know when you may end up working for one of your current subordinates!

FROM "TECHIE" TO MANAGER
(from diodes to dilemmas)

P. Troy Jackson, CDP
Missouri Western State College
Computer Center LRC 104
4525 Downs Drive,
Saint Joseph, Mo. 64507
(816) 271-4565

"When I was a technician, we were real technicians. We trouble shot down to the component, sometimes down to the molecules. And when we repaired something, we didn't just replace a faulty module. No, we actually rebuilt the broken part. And we didn't have all the fancy tools either. We had to strip wire with our teeth and test for a hot lead by licking our fingers and feeling around. Yeah, those were the days I'll tell you."

Sound familiar? Well, when I first started working as a technician in the Navy, there were a few "old salts" who talked like that. I was told how easy us youngsters had it, because we had crimp connectors instead of soldered ones. As silly as some of those old sea stories were, we did learn how to trouble shoot (sometimes down to the component), and how to analyze a circuit in order to isolate the faulty module. More importantly, some of us learned to enjoy doing it. There is quite a special feeling of accomplishment that comes from fixing something.

Now I spend my time sitting behind a desk or watching other technicians perform those tasks. Sometimes I miss the thrill of fixing a broken drive, of seeing a job completed. Now I am a manager who seldom sees a job completed, because management involves working with intangibles. Is it worth the hassle? Yes!! In all honesty, I know I could never be happy as a technician again. I could perform the job, but I wouldn't find it challenging. This paper will address why techs frequently become managers, the differences and similarities of the two professions, some of the pitfalls, and some of the rewards. This is a journey that I not only have made myself, but have helped others make. I hope you can benefit from those experiences.

WHY TECHNICIANS?

Why are technicians so frequently chosen to become managers? With the advent of Management Information Systems (MIS), Decision Support Systems (DSS) and Computer Aided Manufacturing (CAM), a good foundation in technology is a valuable asset to a manager. It is often easier to train a technician to be a manager than to train a manager in a highly technical field. There are also a great many commonalities in the two fields.

Similar Requirements.

To be successful as a technician requires many of the same qualities as a successful manager. The ability to systematically analyze a problem is essential in both areas. To be sure, the problems are significantly different, but the techniques involved in solving them are quite similar. A technician must be able to formulate a plan, an ordered method of determining a problem. A manager likewise must have the ability to plan and organize. Frequently technicians, especially field technicians, must work well with people. They must be able to effectively communicate complicated technical information to non-technical personnel. They must listen well and be able to determine if what is being said is what is actually meant. All these skills are necessary to be a good manager as well.

In addition to the similarities in personality traits, frequently many technical skills are needed by management. As automation becomes more and more commonplace, the need for managers to have a good technical foundation becomes increasingly important. One of the fundamental areas in which a modern manager must be competent is adaptability, the ability to change with new circumstances. Technicians have to master that ability in order to survive in the field. New concepts and methods are constantly appearing and the technician must adapt to those changes or become inefficient. A good manager will find adaptability an invaluable asset in adjusting to the constant undulations of the position.

The Peter Principal.

The Peter Principal is the concept that an individual is promoted to the level of incompetency and remains there. It is rampant in government agencies. Many times a technician is doing an outstanding job, and to "reward" this excellent worker he/she is promoted to a higher level. This continues until the person is no longer excellent, in fact often not even adequate for the task. If a technician has the skills to be a manager, reaching incompetency can be avoided. That doesn't mean they want to be a manager, though, and if someone is unhappy in their position, they will not perform well.

An associate of mine is a good example of the Peter Principal at work. He was a "real" programmer. He loved programming almost more than anything else, and of course he was extremely good at it. As a result of his performance, he was promoted repeatedly until eventually he moved into management. Now he hated managing. He didn't particularly like people and he missed being able to program. Not only was he doing a job he didn't like, he was prevented from doing what he did enjoy. Even though he hated the work, he was competent enough to "get by" so he remained in the position and miserable for several years. Eventually he quit and went to work for another company as a programmer. The company not only lost a valuable employee, they lost him to their competition. Unfortunately my associate is not an isolated

case. This type of thing happens all the time. The common belief is that the career path always leads to management, and declining a promotion is often interpreted as a lack of initiative.

DO YOU WANT TO BE A MANAGER?

This is an important question, one that should be asked before accepting a promotion. Do you have the skills and ability to be a manager? If not, can you develop them? What are your personal feelings? If you decide to enter the management arena, it is important to know what is in store for you there.

Know What You are Getting into.

Earlier I mentioned some of the similarities between managers and technicians. There are also some significant differences. A manager must work with many intangibles and inconsistences. A technician has clear-cut, well-defined problems to deal with. The disk drive won't work, the com link has failed, the serial port is down, it's broken.... A manager's problems are seldom well-defined. Productivity is down, moral is low, absenteeism is increasing, the new hire doesn't "fit in".... As a technician, the same solution always solves similar problems. If a bad diode caused one drive to go down, replacing that diode will probably fix the next drive that fails. A manager does not have the luxury of making that assumption. Just because the last time absenteeism increased and was corrected by simply generating a memo restating the policy on attendance is no indication it will work this time. In fact, as often as not the previous solution is indirectly responsible for the current problem.

On of the most frustrating aspects of being a manager is being responsible for things over which you do not have direct control. If a technician is responsible for repairing a piece of equipment, he/she has complete control over accomplishing that task. Managers must accomplish many of their tasks through others, but the responsibility is still the managers'. Oddly enough, the opposite is also true and just as frustrating. Many managers' problems are caused by their own actions or lack of action. Technicians seldom cause their own problems. They solve problems created by someone else. Managers, on the other hand, create many of their own problems, especially "people problems" like productivity, morale, and absenteeism."

One of the most common reasons people enter into management is the mistaken belief that it automatically pays more. Don't be deceived by the increase in pay. Money can't buy happiness (but then again neither can poverty). If you look only at the actual take home pay, managers usually are paid better. But that's not the whole story. When I first started working at one job, I noticed that my time card was white while most of the others were yellow. I asked the employee relations person about this, and she replied, "oh,

white is for management. It means you can work as many hours as necessary, and you won't mess up your pay check with overtime." As a manager, it is normal that you will be expected to work more hours and be "on-call" when you're not at work.

It's Super Bowl Sunday. You've just settled down in your favorite chair with your favorite beverage in one hand and a bowl of munchies on the table by the other. There is three feet of freshly fallen snow outside, but you're comfortable next to the roaring fire. About that time, the phone rings. A problem has come up at work and you're "in charge." If you're a technician, even if you can't get out of it by responding "numero es wrongo, no hable Englas akee," at least you know you'll get double time for the job. If you're a manager, you'll be lucky if you get a thank you. More likely, you will get you tail chewed for "allowing it to happen."

In addition to working and being "on-call," technicians usually can't take their work home with them. Managers can't avoid it. Frequently, I have laid awake nights contemplating some particularly difficult problem. Admittedly, I have often come up with my best solutions at 2:00 in the morning, but there is no way to avoid bringing your job home. When a technician takes a week or two vacation, the work continues to get done. When a manager takes a week or two vacation, the work accumulates (I sometimes think it actually reproduces in my in box). Even if the work is done by an assistant, the responsibility remains with the manager. My wife chides me because I continually call the office when we are on vacation. If something happens, I am responsible even if I am not there. Part of my job is to insure that things run smoothly when I am gone. If the challenge of developing plans that work even in your absence is something you enjoy, management may be just the thing for you. If you feel better dealing with something you can put your hands on, you better reconsider.

How To Decline A Promotion (and still keep your job).

What if you decide that you really don't want to be a manager? Or worse yet, discover that you're not any good at it? Well, you could do like "the worst manager you ever had the misfortune to work for" did and do it anyway. You could "just say no," but you might be understood as saying you have no initiative. If you don't want to be a manager, it is important that you realize you will probably not be a good manager. It does not help your company or yourself to accept a promotion into management if you will not be happy.

If you are fortunate enough to have someone as a manager who enjoys being a manager, the chances are he/she will understand. I have had the occasion to make the management offer to several persons. I explain what will be expected and why I think that individual is capable of doing a good job. I am batting about .500. Some people are not comfortable in management. Forcing them will defeat the purpose. Not everyone is cut out to be a manager, and that does not imply

they have no desire to advance or excel in their field. Often it is a desire to excel in their field that motivates them to stay in that field.

Not everyone is fortunate. Try to understand. Most managers do not actively seek persons to promote, especially into management. Sometimes they fear they are training their replacement. Sometimes they don't feel comfortable in training a manager or in their ability to do so. Sometimes they feel "it's not my job."

(an aside to managers)

To those of you who might have some of the above feelings, let me explain a few "facts of life." One, if no one else can do what you do, the company cannot afford to promote you. No matter how much more valuable your talents could be if applied elsewhere in the organization, you can't move on until someone is trained to take your place. Two, the same applies to vacations and sick leave. They couldn't get by without you then either. Three, if you are developing a replacement, the company is much better off if, God forbid, you get hit by a truck. Four, most managers can manage, but only a rare few can train good managers. That makes someone who can serve as a mentor a valuable asset, instead of an obstacle. Think about it.

If your manager has not been enlightened, you may have some difficulty explaining why you are not interested. However, it is important to remember that, regardless of how it is taken, you are better off declining the promotion than moving into a field where you don't want to be. Another warning, don't take the job on a "trial basis." If you perform any useful function in your present position, the company cannot afford to leave it open while you "try your hand at management." Also, don't even think you can do both. It cannot be done. Both jobs will suffer, not to mention your sanity.

WHAT MAKES A GOOD MANAGER?

Endless studies have been done to determine the qualities that make a good manager. Some argue, quite eloquently, that they change with circumstances. I am not so presumptuous as to assume that I know <u>everything</u> about being a good manager (especially since humility is a good management trait). However, there are some areas that bear pointing out. The best listing of the qualities of a good manager I have ever seen came from a poem I memorized as a child.

If you can keep your head when all about you
Are losing theirs and blaming it on you,
If you can trust yourself when all men doubt you,
But make allowance for their doubting too;
If you can wait and not be tired by waiting,
Or, being lied about, don't deal in lies,

Or being hated don't give way to hating,
And yet don't look too good, nor talk too wise:

If you can dream--and not make dreams your master;
If you can think--and not make thoughts your aim,
If you can meet with Triumph and Disaster
And treat those two impostors just the same;
If you can bear to hear the truth you've spoken
Twisted by knaves to make a trap for fools,
Or watch the things you gave your life to, broken,
And stoop and build 'em up with worn-out tools:

If you can make one heap of all your winnings
And risk it on one turn of pitch-and-toss,
And lose, and start again at your beginnings,
And never breathe a word about your loss;
If you can force your heart and nerve and sinew
To serve your turn long after they are gone,
And so hold on when there is nothing in you
Except the Will which says to them: "Hold on!"

If you can talk with crowds and keep your virtue,
Or walk with Kings--nor lose the common touch,
If neither foes nor loving friends can hurt you,
If all men count with you, but none too much;
If you can fill the unforgiving minute
With sixty seconds' worth of distance run,
Yours is the Earth and everything that's in it,
And--which is more--you'll be a Man(ager), my son!

("If" by Rudyard Kipling)

Your Team.

One of the most fundamental characteristics of a good manager is the realization that those who work with you are what make you or break you. A friend of mine use to say, "It's not your boss that promotes you, and it's not your boss who fires you. Your people promote you, and your people fire you. Your boss just brings you the news." A manager, by definition, accomplishes most tasks through others. Once that is realized, you begin to think of those who work with you as a team. The manager may be the leader, but the people you work with make things happen. With this understanding comes the ability to realize and praise accomplishments, and that leads to a better team.

Remember, all the things that you use to want your manager to do? To recognize when you did a good job and not only when you messed up. To tell you what they expected and why, instead of just assuming you knew. Motivating people is not much more than "doing unto others as you would have done unto you." I realize that there are cultural, ethnic, and sexual differences, but basically everyone wants to be treated the same. A little respect, a little appreciation, a little recognition go a long, long way. Every good technician knows if you mistreat your tools, you'll lose them. Every good

manager should realize the same holds true for the people who work with you.

Good Managers Are Made, Not Born.

Technicians are made, not born too, but no one argues that. It is intuitively obvious to the most casual disinterested observer that technicians must be trained, but people believe that management skills are somehow second nature. Does that mean you have to have an MBA before you can manage? Of course not! It does mean that you have to know quite a bit about human nature. Just like a technician has to know how his/her tools work to get the most out of them, a manager must know how people work to get the most out of them. Fortunately, one of the best learning areas is life.

Become aware of people. What makes them want to do things? During my MBA studies, it never ceased to amaze me how many studies had been conducted before it was determined that money didn't motivate. It seemed obvious to me that simply giving someone more money wouldn't necessarily make them work any harder. In fact, I've always thought it worked just the opposite. If I did an outstanding job, I would be paid more because I was worth more. Granted, there have been situations where I was "motivated" to look elsewhere for work because I wasn't paid enough, but that was primarily because my manager equated recognition and pay, not because I did. If you want to motivate someone to do a "good job," let them know when they do a good job. Before I went into management, I knew that fundamental truth. It's strange how many people forget it once they become a manager.

I don't mean to imply that <u>all</u> management skills are "common sense." They are not. But a good many are simply a matter of working well with people. All the quantitative analysis and Decision Support Systems ever devised will not make you a good manager if you don't understand people. You can learn Markov analysis and cost analysis, but if you don't communicate with those people who actually get the job done, you're not a good manager. And if you don't get the job done through people, you're not <u>really</u> a manager at all.

PITFALLS

If You Want a Job Done Right ....

In training technicians to be managers, the misconception I encounter most often is that they must do the job personally. By definition, a manager accomplishes tasks through others. The ability to delegate is an absolute necessity for a manager. A technician is judged by how well a task is performed, and it is often necessary to "do the job yourself if you want it done right." A manager that falls into this trap will find they either have an impossibly large amount of work to do, the other employees have little or nothing to do, or both.

One manager I trained had this problem in the extreme. She felt she had to do every job herself in order to insure it was done correctly. I began assigning her increasingly larger work loads. She was an outstanding worker and accepted the increase amount of work and still did the job herself. I believe in monitoring our team in order to "catch" someone doing good so I can encourage them to keep it up. I decided that it wasn't "good" to have a manager who couldn't or wouldn't delegate, so I inflicted the ultimate punishment. I stopped telling the manager she was doing an outstanding job. The tasks were being accomplished quite well and she was working very hard, but she was not doing her job. She was doing the job of her teammates and preventing them from succeeding. I told her this and began noticing and complimenting her every time she did delegate. I even went so far to compliment her when she asked someone to get her a screwdriver so she could tighten a connection. In a few weeks, she could delegate with the best of them. Now she very judiciously assigns tasks, supervises their completion, encourages and recognizes those who work with her, and, in short, is becoming an outstanding manager.

It is important to remember, however, that you can only delegate the task, not the responsibility for the task. Even though a manager delegates a job to another, the responsibility remains his/hers. If a function is not completed, who should be held responsible? Wasn't it the manager's duty to correctly assign the tasks? Only the manager has the duty of assigning the jobs. Only the manager has the responsibility of assuring they are completed. Managers must delegate, but they also must be cognizant that the ultimate responsibility remains theirs.

Give Credit Where Credit is Due.

Some new managers appear to be afraid that they will lose their position if they recognize the achievements of their teammates. This is a ridiculous notion and a common pitfall. If you just ignore your team, they'll go away. A technician should know everything they can about the equipment they work with. A manager should know everything possible about managing, but not everything about everything. As mentioned

earlier, technicians are sometimes "rewarded" for their knowledge by moving them into management. They begin to think they can succeed only if they know more than the other technicians. If you apply some technician's logic to this, the fallacy becomes readily apparent.

If all the files had to be maintained on the system disk as well as each of the subsequent drives, not only would the system disk be huge, but it would take forever to get anything done. The same is true in management. There is no shame in not knowing everything. The shame is the manager who refuses to admit it. Let your team work. They, not their tasks, are your job. The better you team looks, the better you look. One of the few things worse than not giving credit is taking credit for someone else's accomplishments. Few people are so crass as to blatantly do this. (Although I once had a director who asked me to put his name on a report I wrote.) But allowing someone to think you did a task that one of your other team members did is tantamount to the same thing. It is far better for everyone to raise the standard than to climb over someone else. If the team looks bad, the manager looks bad. You would be surprised at how far a little praise and recognition goes. People who are certain their manager will give them credit will do more, and isn't that what the manager is trying to accomplish?

Power Corrupts

"The job went right to his head." That attitude is indicative of a dissatisfied co-worker. Some people have a hard time handling authority. Sometimes it stems from not being able to delegate, sometimes from a fear of looking bad, and sometimes because of psychological problems, but the results are the same. Rudy "tech" becomes Rudy "manager" and suddenly, overnight, it's Dr. Jekyll and Mr. Hyde. Friends disappear. Old habits change. Instead of shooting the bull with the guys, it's "back to work, you dummies." In many cases, it is a matter of perception. Some things must change when a tech becomes a manager, and there is bound to be some animosity, especially if it is perceived that he/she doesn't know what is going on. But a good manager knows his/her limitations and works on improving them. The realization that the team is what counts and doing the best job to accomplish the goals of the organization helps everyone. If the product doesn't sell, no one can get a raise. If the company doesn't succeed, everyone loses.

A CHANGE OF PERSPECTIVE

There are a great many changes that a technician must go through to become a manager. One of the hardest for those around you to accept is the change of attitude that must take place. You must learn to view everything a little differently than before. You wear a suit to work, not because a policy says so, but because you represent a company and image is important. You must look out for the company's best interest. Most non-managers, or poor managers, see this as a dichotomy

when compared with what I said earlier. How can you become a "company man" and still be concerned with people? It's actually quite simple. What is a company but people? It's this broader perspective that is difficult for some people to grasp. It stems from the adversarial roles that have commonly been assumed by "labor" and "management." Someone has to lose in order for someone else to win. A good manager must realize that only when the company benefits can the individuals that comprise the company benefit.

Let's look at an example. Some cost efficient person determines that it will save $500 a year if the free coffee in the employee lounge is the "plain label" brand instead of the good stuff, and the company can make another $500 a year if they charge ten cents a cup. What does the good manager do? The mediocre manager realizes that the employees are going to be upset if the company starts charging, but will accept the switch to the cheaper brand. The smart manager understands that the $500 saved will be lost in reduced productivity, and so does not change. The outstanding manager is the one, who after seeing a higher-up make the change, can recognize the potential for losses to the company, circumvent the implementation of the plan, prepare and present the issue to his/her superior, and persuade that person to reverse the planned change before it is ever announced. (Bonus points if you can somehow turn it into an opportunity to increase productivity.)

The emphasis of this paper has been on the transition from technician to manager. Usually there is an interim step called supervisor, but sometimes there isn't. A large part of management is coordinating and dealing with people, but that's not all folks. There are a plethora of differences in going from a "blue collar to white collar." But don't get so wrapped up in "climbing the corporate ladder" that you forget who holds it up. Eventually the tools or program listings that used to fill your briefcase will be replaced with concepts and plans and budgets. It's not an easy transition, and there is a great deal that must be learned. Not only must you be able to come up with good ideas, but you must be capable of presenting them so that they will be accepted. The rigors of this path are many, but so are the rewards.

WHY BOTHER?

I have spent the majority of this report maligning the problems and difficulties of management. I would be remiss if I failed to mention the benefits. During a particularly difficult period in a recent relocation (yes, this is a plug for my other paper), I was asked by someone I was training why I "put up" with all the hassles. It was during one of the rare moments that seem to lend themselves to reflection, so I considered my answer carefully before I gave it. "I can't think of anything else I enjoy doing enough to put up with the hassles" was my reply. Granted, it takes a certain "glutton for punishment" kind of personality to enjoy this type of work, but that's part of the challenge. To a large

degree, you control your own destiny. Short of owning a company, where else can you find that degree of independence? Sometimes you have to "play the game," but that's part of the challenge too. Not only do you have to know what's right, you must be able to convince others. To be sure, it isn't easy... but what is, that's worth doing? If the idea of taking a concept, a dream, and molding it into reality, within the constraints of an organization and through other people does not appeal to you, then don't go into management. If it does, I wish you all the luck. It's nice to find a kindred spirit.

"Behold the turtle. He never goes anywhere without sticking his neck out."

Think about it.

The Greeting Behavior of the North American Homo Sapiens

Louis R. Mills, CDP

Bio-Rad Laboratories, Inc.
1000 Alfred Nobel Drive
Hercules, CA 94547

(415) 724-5613

Most of you don't know how to greet strangers. In fact, most of you don't even know what it is you do when you meet someone. Many of you have your minds turned off. Most of you don't realize that greeting behavior is culturally defined and often situational.

Have you ever gone to work when you've been sick, feverish, aching all over and generally feeling sorry for yourself? And you're walking down the hallway, wishing you were somewhere else, when a co-worker walks past and says "Hi, how are you?"

98% of you will reply "I'm fine!" That, ladies and gentlemen, is a patterned response. A "habit" that is so ingrained, that we react in a specific manner to a specific trigger. Just like Pavlov's dogs.

We do it with friends, we do it with strangers. We never took a class in Greeting Behavior, we never laid out ten different greeting styles to fit all ocassions. (Except for those of us who were awkward teenagers trying to figure out how to ask for that first date.) And we usually don't think about altering our style of greeting.

But we've all heard how important that first impression is. Haven't you noticed that some people turn you off the minute you meet them? And others you fall in love with almost immediately? Well, a lot of that reaction has to do with their greeting style, and a lot of it has to do with what's going on in your head.

There is a psychological and physiological basis for a lot of what happens during the "greeting" process. In fact, psychologists break our behavior with others into three phases:

1. Contact (the Greeting period of up to four minutes)
2. Rapport
3. Parting

Today, we are going to minutely examine that first phase, become aware of some of the mental and physical things that happen to us during it, and explore some alternatives.

First, let's take a simple case. Let's look at animal greeting behavior:

1. Dogs

2. Cats

What was primitive man like? Probably like the apes and baboons.

What are the human dynamics?

1. Sight. 83% of learning via this sense.

2. Hearing. 11% of learning.

3. Smell/Taste. Subtle, but powerful influence. Shortest neural path to the brain. Most primitive of the senses.

4. Touch. Touching can be "violating". Personal space is important.

5. Conceptualizations. "Pigeon-holing"

What are the situational dynamics?

1. Formal, strangers.

2. Formal, acquaintances.

3. Informal.

4. Intimate.

Let's explore some examples of these human and situational dynamics.

With each of these, the whole range of emotional and physical reactions are subtly different. Yet many people create poor impressions or "give off bad vibes". The cause may be:

1. Appearance. Facial features, weight, clothing. Eye contact is culturally defined.

2. Voice. Deep, squeaky, monotonous, breathy.

3. Smell. "He stinks". Cologne and perfume.

4. Touch. Handshake quality, space intrusion.

5. Our minds. Similarities to others. Prejudices for and against other groups. Previously defined stature or "past". Prominence vs. "Low life"

How does <u>our</u> attitude affect our greeting behavior? Well, imagine that the next person you meet, friend or stranger, might fall into one of these categories:

1. The Love of Your Life!

2. Your next boss.

3. The Mentor who changes your life.

How can you become more sensitized to the way you look?

1. Mirror(s)

2. Photograph(s)

3. The "Private Eye"

What else can you do to make a better impression?

1. Eye contact

2. Speak slowly and clearly.

3. Repeat the other's name.

4. Learn to shake hands.

What constitutes the perfect greeting?

1. Looking good
2. Correct amount of eye contact
3. Firm, short handshake
4. Use the other's name
5. Smile
6. Relax
7. Pay a sincere compliment

Practice what you've learned.

TELEPHONE TRICKS TO AVOID TELEPHONE TERROR

Karen L. Jackson
Computer Center, LRC 104
Missouri Western State College
4525 Downs Drive
Saint Joseph, Missouri 64507
(816) 233-1021 or 271-4565

Are you one of those people who dreads having to make a phone call? Do you have a hard time doing business with people you cannot see face to face? For the most part, I have always been at ease using the telephone. In fact, I was one of those teenagers whose parents wondered if the phone was permanently attached to my ear. There have been times, however, when I have had to struggle with the many excuses and force myself to make an unpleasant call. For about four years, I provided phone support and training for a software company. I gave customers good news and bad news as well as straightforward answers. I got to listen to them when they had been up all night or maybe even all weekend trying to solve some very illusive problem that may or may not be our fault. Once I even got to call a dozen or so customers and tell them not to install the long-awaited software distribution we just shipped because it still had a serious bug in it. There was no way I could do my job without the telephone. Alexander Graham Bell invented a device that is very useful to us in business today, and I hope to help you make better use of it in your life.

First, lets take a look at the telephone and some of the ways it is used. It is a communications tool. It allows us to speak with another person in the next room, the next block, or around the world. The telephone can also be used to communicate in other ways, such as transmitting data, pictures, etc. I will focus primarily on the original use for interpersonal conversation. The telephone is used extensively both in business and in social communication.

TYPICAL TELEPHONE TERRORISM

Have you ever wondered why you avoid the telephone to the extent you do? There are many possible reasons. Perhaps you rely on body language when you communicate, both to send messages and to receive messages. Eye contact is very important to most people in carrying on a conversation. It is very distracting to be speaking to someone face to face and have their eyes wander about the room. It is an indicator of whether or not they are listening to you. I find it much easier to talk on the phone with someone I have previously met. Having seen them personally, I am able to visualize what they look like while I am talking to them on the phone. I find myself imagining their body movements in response to what I am telling them. This visualization gives me the feeling of dealing with a real person, which is what I am doing. I

am just using some electronics as an intermediary. And this way, if I make them mad, I don't get the full force of their blow either.

A hearing problem can also cause you to avoid using the phone. Sometimes people supplement poor hearing with lip reading without even being aware they are doing it. Perhaps your hearing should be checked. Poor telephones or phone systems can also make hearing very difficult. Background noise, such as a printer in your office can be a cause of avoiding the telephone. Fixing the phone, getting a hearing aid or amplified telephone, and moving the noisy printer could make the telephone more useful to you.

Do you consider the telephone an insult? Maybe it seems demanding and impatient when you have important things on your mind. When I was programming, a brief interruption by a phone call could set me back a full hour in my programming. Sometimes it was urgent, but other times it was trivial. I did not have a secretary to answer my phone and screen my calls, so I felt it was necessary to answer it myself. It might be that you find it demeaning as well as distracting to have to answer your own phone.

TEN TRICKS (more or less) TO TAME TERROR

Organization is a key to efficient use of the telephone. "Be prepared," to quote the Boy Scouts. Before you dial, take time to think about what it is you intend to accomplish with this call. Prepare the questions you intend to ask. Make some notes. Think about what questions you will be asked and about the answers to those questions. This reminds me of an experience of one of my co-workers. In her previous employment she received a call from a secretary who stated that "I pushed the button and the computer is doing funny things." When Donna asked her to explain that a little, she repeated her first statement verbatim. Donna then asked her which button she was pushing, and the secretary replied, "I knew you couldn't help me," while slamming down the phone. A little help from this lady would have allowed Donna to begin isolating and then solving the problem. If you call a repairman to fix your television, you usually need to know the brand and model or serial number. Frequently you are asked about what it is doing or not doing. If you take the time to gather this information first, then you are helping yourself as well as the person on the other end of the line. This applies equally to business situations.

During your conversation, take notes. In the course of providing software support, I frequently had to talk with our systems programmers to find the answers for the customers. Even though I had been a programmer for several years, these people could snow me. There is nothing wrong with asking someone to wait a moment while you write it down. It is far more efficient to get it correct the first time, than to call back a few minutes later because you forgot part of what they said. After you write it down, verify what you have written.

Read it back and ask the other person to correct you if necessary. I learned this from a boss I had several years ago. He told me that if I ever needed to leave a message for him at home and one of his kids took the message, I should verify it. Do you have a pencil? Do you have paper? Now read back what you wrote down. My husband and I are still trying to convince our own teenagers of the need to get message straight. It is even more important in business that messages are taken correctly. That female voice on the other end that you assume to be an experienced secretary who knows what she is doing and routinely takes messages could be anyone. It could be the boss of the person you are calling, a little irate because Joe is nowhere around and she got stuck answering his phone again. It could be a student intern, nervous and intimidated by talking to a lot of very important business people. I remember an incident that occurred to me a few years back. I was a newly hired faculty member at a college in Nebraska, and I had the "privilege" of serving on the screening committee to fill an administrative position. I had to do reference checks on the applicants. This meant I was questioning college presidents. That was intimidation. There are tactful ways of making sure the message is correct, such as "do you need me to repeat any of that?", or "could you read that back to me so I can make sure I didn't forget anything?" Take a moment just to be sure.

At the end of the call, summarize what has transpired. Make sure that both you and the other party are clear as to what is to follow. Have you ever sat and fumed and waited for someone to call you back? Eventually you call them or they call you only to find that they were sitting and fuming and waiting for you to call them back. "I should expect to hear from you tomorrow morning, but if I don't, I will call you as soon as I return from lunch." "I will call you first thing Monday morning, but sometimes I get distracted. So give me a call if you haven't heard from me by nine." "You will not get your final payment until you have delivered a printer that works." Any of these makes it perfectly clear what the next step is and who is supposed to make the next move.

Sometimes it is necessary to follow up a phone call with a letter. That last example about withholding payment is one of those situations. If you are giving a vendor an ultimatum regarding inadequate service, it is okay to begin with a phone call; however it should be immediately followed by a letter. If someone promises you something, get it in writing, either by having them send you a letter or by writing a letter of confirmation yourself. A phone call can get the ball rolling, but a letter may be necessary to keep the ball rolling.

The telephone can be a nuisance too. Junk calls have reached the same level of annoyance as junk mail. Having a secretary screen your calls serves several purposes. First of all, it can keep you from spending endless minutes, which add up to hours, talking to some salesperson who will not hang up. Second, it allows you to prepare to talk to the person. It allows you to make notes and prepare questions as for any

outgoing call. Have the secretary get their name, number, what the call is regarding, and when would be a good time to reach them. Should one of those junk calls get through to you, beware. Some of them are more than annoyances. Some are simply scams. The best way to deal with these is to end them as soon as possible. Keeping in mind that phone solicitors are trained to keep you on the phone, this is not always easy. I found it very satisfactory to simply say that I am not interested, thank you, while I am hanging up the phone. It doesn't matter if they continue to talk; I just hang up. I don't feel guilty because I have been rude (even though they may deserve it), and I have never had one of them call me back.

A good receptionist is worth his/her weight in gold. Their ability to screen out nuisance calls, to recognize the fake old buddy, good friend manner of a salesperson is not the limit of it. They can recognize who is really important and who is just important. They can also direct calls to right person in the first place. Most of my years in software support, I worked as a team with a fellow named Mike. He and I complemented each other in knowledge and experience, but we both were very capable in our jobs. We had one customer who, whenever a female voice answered, demanded to "talk to someone who knows something." Needless to say, whatever I told him was ignored. Our receptionist usually directed his calls to Mike, who would proceed to ask me for the answer to questions in my realm of expertise. This customer went so far as to ask me why I wasn't staying home and having kids. If only he had known that I was sitting there eight months pregnant, shoes kicked off, and feet up on my desk...the stereotypical barefoot and pregnant. There are many other "personal preferences" a good receptionist can handle.

Time is of the essence. This applies to phone calls too, especially when the customer base you are dealing with is international. If you live in Missouri, you have Nevada, Mexico, Oregon, Paris, Moscow, Savannah, and many other familiar names all within the state. When most people call London or Sydney from the United States, they are crossing many time zones. Some clients were leaving for the day before I even had my morning coffee break. Others would be arriving for tomorrow before I was done for today. Canadian clients did not observe daylight savings time. In order to return calls and reach customers, it was necessary to keep track of these many differences. I certainly made use of my high school geography knowledge, at least what little was left of it. If you regularly do business across time zones, consider a clock with multiple time zones or even multiple clocks. Take notes and label your notes with the time difference. World wide holidays vary too. Canadian Thanksgiving is in early October. Of course everyone knows that the day after Christmas is Boxing Day (but when is Wrestling Day?) and the entire United Kingdom has a holiday. I can't remember when tea time is, but I think it is before (or is it after?) siesta time. If you are dealing with a "real programmer" you never know when they will be at their desk.

Unless you are different from most, you probably find it very annoying and frustrating to spend what seems like an eternity on hold listening to someone else's idea of pleasant, relaxing music (My pediatrician chose rock music, which I found intolerable with a sick baby screaming in the other ear). When you finally get through, inquire as to when the slow time of the day, week, month, etc is. When an immediate reply is not mandatory, you can call during a slack time. Not only do you avoid the wait on hold, but the person on the other end is usually more patient and more helpful when not so hassled. Find out if this is a nine to five person, a noon to eight person, or a four to noon person. Place your calls accordingly.

TRANSCRIBING TELEPHONE TIDINGS

Leaving and taking good messages is another key to effective use of the telephone. As I mentioned earlier, getting the message written down is important, but it is also critical that all the necessary information be included. Who is the call for? When can the person be reached? What is the call regarding? What company are you/they with? Phone tag is not a fun game to play. Answering machine tag is even less fun to play. The older answering machines limited the length of the message so much that it has been the subject of many jokes. Everybody seems to know what they want to receive in a message, but they seldom remember it when leaving a message. We have a machine at home, and it is most annoying to have message that says, "call me back." There are eight people in our family. I don't recognize the voice, and neither does anyone else. Needless to say, we don't call back. Another problem is when you have been away for a day or two, and the message says Bill will be there until ten. You don't know if Bill meant today or when. Written messages nearly always contain the date and time of the call, because most businesses use forms with a blank for them. Machine messages seldom do. Make sure your machine messages are complete and intelligible. Talk a little slower than normal, and enunciate clearly.

TELEPHONE TEAM TALKING AND OTHER TECHNOLOGY

Conference calling allows several people to be linked together for a telephone conference. Cost-wise, this is a very effective way of discussing business without having everyone fly in from wherever. It can be very awkward. My husband was interviewed for a job this way, and he felt a little uncomfortable discussing his skills so thoroughly with people he had never even met. Actually, it was a preliminary interview, but it lasted for well over an hour. It allowed the company to do a better job of screening the applicants from around the country before selecting a few for on-site interviews. Many times, in dealing with some highly technical software problems that I didn't understand, the customer would talk directly with the systems people and I would listen in. By doing this I could learn what questions to ask if a similar situation should arise in the future, and I could also learn

what the answer was. It was highly informative. Out of courtesy, I always asked both parties first so they knew I would be eavesdropping.

Speed dialing can simplify using the telephone. By coding in the numbers of your most frequent calls, you can place calls by dialing an abbreviated number. It also reduces the chance of reaching a wrong number. Some long distance phone companies require the use of lengthy codes and these too can included in your speed dialing numbers. With speed dialing, you normally program in the numbers you want, assigning each a one to three digit code, depending upon the system you have. When dialing you simply dial the short code and a # or *. A long distance code can be inserted in the middle when dialing with most systems. Speed dialing can be implemented through most phone companies, your own telephone, or add-on equipment.

Adding a fax machine to phone system can enhance its usefulness significantly. With a fax you transmit pictures or documents. You know the old saying, a picture is worth a thousand words. Many needlessly long conversations can be shortened by faxing a document, diagram, or picture. If you do not have a fax machine in your business, "public" machines are becoming more and more common. Many hotels, package mailing services, libraries, and other businesses have fee for use machines available.

Cellular phones in your car can be useful if you spend a lot of time on the road, whether covering distances or stuck in urban traffic jams. They allow mobile communication far better than was available not so many years back and serve a useful purpose other than just being a yuppie status symbol. There is some expense involved, but it may well be justifiable. If you think this could be of use to you, check out what is available through a local vendor.

Your telephone system may not be state of the art technology, but it may be capable of doing a lot more than you realize. Look into finding out about the features available to you at no extra charge. Can you forward calls, either to keep your phone from interrupting you or just to have someone answer it while you are away? Do have some type of speed dialing and not even know about it? Communications technology is changing rapidly and has become much more competitive since the breakup of Ma Bell. At the same time, we no longer can expect our local phone company to do it all for us. We need to go out, look around, check out what is available, and do what we can to see that the phone system meets our needs in a cost effective manner.

With some of the new technology such as ISDN, some innovative features are becoming available. Screening calls becomes easier when you can see the number of the person calling before even answering the phone. Simply don't answer

the phone. Statistical analysis of both incoming and outgoing calls can be made, allowing optimization of WATS-type lines to most closely meet your needs.

TYPICAL TELEPHONE TALK

In conclusion, I want to take a look at some typical conversations and discuss how they could improved upon. For the first example, let's assume that Pete is calling someone to provide software support to his company. He has a contract with them to provide this service, so he is identifiable to them as a customer.

Jim: Hello. Customer support, this is Jim.

Pete: Hello. I, uh, I'm having a problem with my Super Simple Software System.

Jim: What company are you with?

Pete: Programmer Pete's.

Jim: And to whom am I speaking?

Pete: Pete Peterson.

Jim: Okay, Mr. Peterson, which of your systems is giving you trouble?

Pete: The big one.

Jim: And what model is that?

Pete: The HP three thousand seventy.

Jim: The HP 3000 series 70. What version of Super Simple Software are you running on it?

Pete: I dunno...just a minute and I'll check. [a few minutes later] Are you still there, Bill?

Jim: Yes, but it is Jim you are talking to.

Pete: They said we have version 4.0, but they haven't installed it yet. So we are running whatever came out before that, I guess.

Jim: Could you describe the problem to me?

Pete: [Pete describes the problem.]

Jim: That was a bug prior to version 4.0. If you install your new distribution, that should solve the problem. Is there any reason you can't do that?

Pete: No, we have been meaning to do that since it came in last summer, but we just haven't gotten around to it. We will

do that one of these days.

Jim: Call me back if that doesn't solve it for you. Good bye.

Pete: Good bye.

And now for the condensed version.

Jim: Hello, software support, this is Jim.

Pete: Hello, this is Pete Peterson with Pete's Programmers. [pause to allow Jim to check his data base] We are having some trouble with our Super Simple Software System. We are running version 3.5 on an HP 3000 series 70.

Jim: Tell me more about your problem, Mr. Peterson.

Pete: Well, Jim [he wrote the name down, so he knows whom he has spoken with], ...[he describes the problem]

Jim: That was a bug prior to version 4.0, but it has been fixed in that release. You should have received your distribution some time ago. Do you have it available?

Pete: Yes, we do. We have been very busy, but if that should fix the problem, we should be able to get installed within a week. Can I call you back, Jim, if that doesn't do it?

Jim: Sure can, Mr. Peterson. Good bye.

Pete: Good bye.

If your department is providing support to some other company, it is virtually the same. You are on the receiving end instead.

Finally, let's use the example of a department providing support and training within the company. Susan is head of the accounting department. Carol is head of the System Support group. Information Services (the department Carol is a part of) is installing a new package, adding on, but not replacing some automated functions for the accounting people.

Susan: Hello Carol, this is Susan in accounting. We are getting that new stuff on the computer this week and need to learn how to use by the first, next Wednesday.

Carol: Oh, okay. [No one has bothered to tell Carol about this, but she controls her urge to ask what on earth Susan is talking about.] I guess you need some training real soon. What did you have in mind?

Susan: [Susan replies with some not totally unreasonable ideas.]

Carol: I will need to check the schedules for the training

rooms and check with Elaine. [Carol buys some time to do the homework and preparation for this call she has received totally unexpectedly] Can I call you back?

Susan: Okay. [What Susan neglects to mention is that she plans on leaving in thirty minutes.] Thanks. Bye.

Carol: Bye.

[One hour later]

Joe: Accounting, Susan's desk.

Carol: This is Carol in System Support. Is Susan available?

Joe: No, she is gone for the day. She will not be back until Monday.

Carol: I need to talk with someone regarding the training for the new package. The training rooms are all booked until the tenth.

Joe: She didn't say anything about it. I don't know what to tell you. She went on a short safari in Africa and cannot be reached.

All of this could have been prevented if only Carol had asked if she could call her back at a set time, even an approximate one. Then Carol would have known that Susan was planning on leaving in thirty minutes. Granted, Susan could have done a better job on her end, but you can't directly change what other people do or don't do, only what you do yourself. That is one of reasons for taking notes, to cover yourself in the event of a discrepancy. Of course all of this assumes that the standard games and office politics are at a minimum and do not interfere too much with the desire to get the job done right.

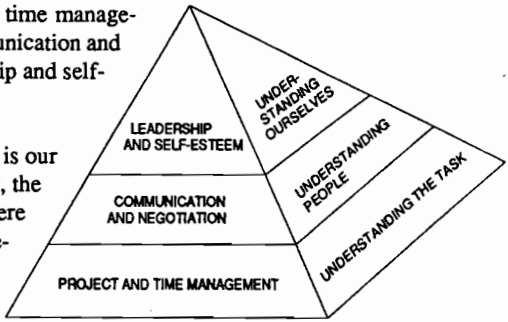
The next time a blizzard is raging and you need to talk to someone in the next building, don't go out in the cold. Pick up that phone. It won't give you frostbite. It might even warm you up a bit when you "reach out and touch someone" via the telephone.

Managing In The New Age

Teresa L. Norman
Tymlabs Corporation
811 Barton Springs Road
Austin, Texas
78704
(512) 478-0611

It has been said by counselor John Bradshaw that we have become human doings rather than human beings. The purpose of this tutorial is to explore the "being" of management and how that affects the "doing" of management. Stated another way, the purpose of this tutorial is to examine how who we are determines our performance in the important managerial tasks of problem solving, communication, and leadership.

Usually, our development as managers follows this course: We learn project and time management; we learn the skills of communication and negotiation; and we learn leadership and self-esteem.



What this development assumes is our growing understanding of the task, the people involved, and ourselves. There aretechniques to help us in this development. The techniques that I have chosen to discuss in this paper fall into three broad categories: problem solving (also called creative thinking), communication, and leadership (also called self-esteem).

Creative Thinking

David Lewis and James Greene in their book *Thinking Better* state, "... we generally find that people who are sufficiently interested to want to find ways of thinking better are among the most motivated and ambitious members of society." However, without the proper tools, even intellectually motivated individuals may not be using their brains as efficiently as possible. Lewis and Greene connect this inefficiency to low self-esteem and anxiety. We can release ourselves from these self-imposed thinking limitations by investing time in a thinking training course such as that outlined in *Thinking Better*. Their techniques deliver the following benefits:

. Ability to make decisions more quickly

. Ability to learn new material more rapidly

. Ability to think more creatively about problems requiring an original approach

. Reduction in mental and physical stress experienced when working under pressure

Edward de Bono, author of numerous books and training courses on the subject of thinking, has said that thought is "our most precious resource". Yet most of us feel uncomfortable saying something like: "You pay me to think. So I am sitting here thinking." De Bono also says: "Anyone can run but athletes are trained to run well." When do we get trained to *think* well? Certainly not in college. In college we are too busy learning about the thinking of others. Most of us could benefit greatly from a little training and exercise for thinking.

In his book *Six Thinking Hats,* de Bono gives us a new way to think about the problems and opportunities we face as managers. His analogy is very simple: a grown up version of putting on your thinking hat. In de Bono's analogy, there are six hats, each a different color. Each hat represents a different attitude, a different perspective.

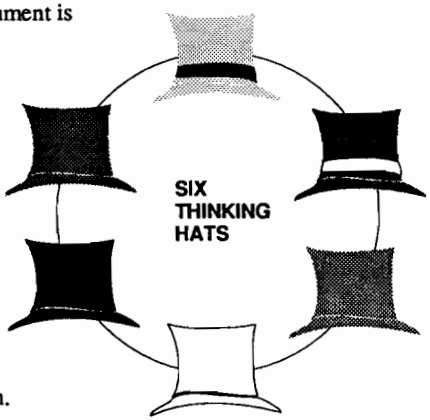
To use de Bono's technique, you think of your problem from each of the six perspectives, represented by the six thinking hats. When you have looked at the situation from six perspectives, you have the entire picture. His technique is analogous to the four-color printing process, where you can't see the complete picture until each separate color has been printed. First you take one approach, one attitude, toward your situation and see how it looks. Then you consciously switch to another attitude, another viewpoint, and see how your situation appears. By stepping through each of the perspectives represented by the six differently colored thinking hats you arrive at a more complete picture of your situation.

Let's look at each hat and what kind of thinking it represents.

White hat thinking

When you use white hat thinking you think unemotionally: just the facts please. You don't evaluate or interpret the information you think neutrally and objectively. You think like a computer.

In Japan, our style of debate and argument is considered rude. A typical Japanese meeting consists of everyone taking their turn stating all facts that they know about the situation. Only then is a decision made. This is white hat thinking.



Red hat thinking

Red hat thinking stresses the importance of listening to your feelings. It is typified by a hunch or intuition, and looks at how you feel about or react to a particular situation.

An important aspect of red hat thinking is that you have no need to justify why you feel the way you do–you simply feel that way. Here is an example of red hat thinking: "I know we should install network X but I *feel* that it won't take our communications into the next decade."

Yellow hat thinking

The yellow hat symbolizes optimism, enthusiasm, positive thinking. Yellow hat thinking is focused on the benefits, not the possible pitfalls. "When this works we are going to make a lot of money." Optimism can create enthusiasm and give a group motivation.

Black hat thinking

This hat represents the opposite of the yellow hat: the logical negative, or pessimism. The black hat is worn by the devil's advocate: "I know that this could be a very successful venture, but will the market be gone before we can produce?"

Assessing the potential negative impact of a course of action is crucial to managing a business. It is much more sensible to look at adverse circumstances in a

planning meeting than to have them forced on us by a profit and loss statement.

Green hat thinking

The green hat symbolizes the new, the random, the creative. When you put on the green hat, you are brainstorming. Any idea, however unusual, can be entertained: "What if we passed out cards with nothing on them but our name?"

Besides creating totally new alternatives, green hat thinking can motivate your thinking from other perspectives: "What if we put up a *billboard* that is blank except for our name?"

Blue hat thinking

Blue hat thinkinglooks at methodology, or what might be called software for thinking. It asks: "What steps do we need to take? Are we being too optimistic? Can we be more creative?" Blue hat thinking is the orchestrator of all the other kinds of thinking. When you use this attitude you are organizing the results of the other approaches. You are deciding which of the other five hats you will wear and in what order.

This technique of de Bono's is a *mental* approach. You probably would never sit down and consciously work through each and every attitude in detail (though that might be very effective). More often you will use this technique implicitly, changing from one attitude to another as you see fit. However, with this technique you will *know* when you are thinking optimistically or pessimistically or rationally.

If this technique seems a little contrived, let me point out the value of this approach to thinking. First, when you refer to the hats metaphor explicitly in your decision making process, you are tapping the power of *role-playing*. Playing a role allows us to experiment, to express and try ideas freely. Second, this approach *focuses our attention* on the various aspects of a situation. Also, this approach *allows criticisms to be shared* in a non-threatening way. We can ask ourselves, or someone else, to be negative or to stop being negative and be emotional, rational, optimistic, etc. When we are accustomed to this approach, we have reliable "rules for the game".

One last word about this technique: As your management skill increases, you will realize that these six approaches to thinking also describe six *personalities*. Who on your staff represents pessimistic thinking? Who can be counted on for optimism and enthusiasm—no matter how risky the prospects? These six approaches help you to understand your staff and how they will react, and can be applied to your own decision making style as well: "I wish I would have thought that one through before I... I was so optimistic."

In summary, de Bono has given us a valuable technique for creative thinking. We can apply it to managerial situations, to understanding our staff, and to understanding ourselves.

Communication

It was amazing! Just when I had gone as far as I could in describing the situation, my associate picked up my thought and continued it for the rest of us. The manager on the other side of the table agreed and pointed out how we could extend the ideas further. Another tied the entire project succinctly into our scheduling while a fifth summed up the entire meeting in under two minutes. We were in total agreement!

Then I woke up.

Let's face it. This is a dream. This does not happen. If it has happened to you—fine. You will have at least one amazing management story for your managerial grandparenthood.

My second topic is communication, specifically communication skills. This information comes to us from a book by Genie Z. LaBorde titled "Influencing with Integrity." Her approach is called Syntonics. Syntonics comes from the word syntony which means to be in harmony—harmony with yourself and others.

LaBorde says that these skills are necessary for excellence in communication:

- **To see and hear more**
- **To adapt to your partners style**
- **To be consistent with yourself**
- **To establish good outcomes**
- **To establish rapport**

Communication is more a matter of perceiving than doing. In other words, it is not so much *what you say* as how you *listen and adapt* that determines how effectively you communicate. So the first skill that LaBorde discusses is the ability to see and hear more.

We have all learned from management training that we must listen more carefully. But there is more to listening then just hearing the words. Seeing and hearing should encompass much more than what is said. For example, LaBorde points out, are we aware of how we are *affecting* our audience? Have we ascertained the concerns that probably lie between the lines of what they say? Have we realized our partner's *style* of communicating?

Our partner's style of communicating is very important. Just ask directions from a New York cab driver and a West Texan to see an extreme example of differing styles of communication. One will be a terse, rapid fire list of streets and landmarks complete with alternate routes for rush hour; the other will probably be a slow, accurate description of not only the best way to get there, but also a few entertaining anecdotes about the people or scenery you are likely to encounter. One will be a hurry-up map of how to go while the other will be infused with the enjoyment of just going.

If your style of communicating is slow and careful, you are likely to irritate the New Yorker who doesn't have time (or even see the need) to spell everything out in detail. If you rush the Texan the directions might lead you to another state! We must learn to adjust our style to our partner's style.

Confusion can ruin excellent communication. And when our words, tones, face, and gestures are out of sync, we confuse our partner. The point that LaBorde makes is that you must say the same thing with your words, your tone of voice, your facial expressions and your body language. If your words express confidence but your facial expressions express doubt, your partner will be confused. They will either have very low confidence in your words or they will just not believe you. What Laborde is telling us is that we must be consistent within ourselves to be excellent communicators. It is not enough to know what we want to say, we must say it with our words, facial expressions, and postures. We must *believe* it!

To create excellent communication, we must all establish good outcomes and rapport. Establishing a good outcome begins by deciding what you want to achieve. A journey begins with a destination. To communicate well, you must select an outcome. Do you know why are you sitting down to talk with your boss, associate, employee, spouse, child or parent? What outcome do you wish to achieve? If you don't know what you want, you are likely to confuse and frustrate them.

Once you know what you want to achieve, you must be careful how you present it to yourself: How have you phrased it? Have you put your outcome in positive terms? Are you saying "I want to be current in all my projects"? Or have you said "I don't want to be behind"? To understand why it is important to phrase your outcome positively, ask yourself this question: How you would *feel* if one of your employees came to you and said "I don't want to be behind anymore. Can you help me?" Now imagine that they said "I want to be current in all my work. How can I do that?" Somehow, for some reason, most everyone is more willing to listen to a positively phrased outcome than to a negatively phrased outcome.

Another part of establishing good outcomes is imagining the outcome. Express your outcome in terms of all your senses: What will I see? (Boss shaking my hand.) What will I hear? ("Yes, you got the raise.") There are two reasons why seeing/hearing/feeling the outcome is important: If you don't know what the place looks like, how will you know when

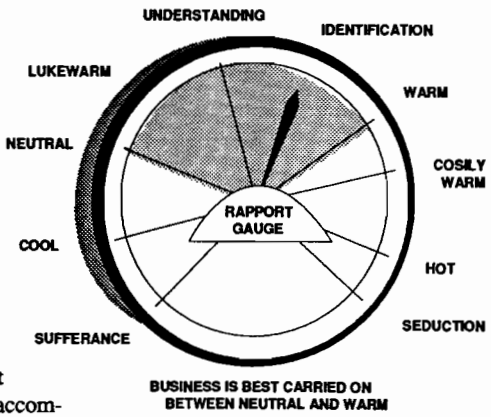
you're there? And, if you know what your outcome is like you can recognize signs along the way that tell you you are going in the right direction.

Another important part of establishing a good outcome is knowing what your partner wants. If you don't understand your partner's concern, you are in for a surprise. Imagine this situation: You ask for a raise but you don't know that your boss has just reviewed a very unfavorable financial report. Surprise! No raise! It probably wouldn't even matter that the report was for another division. But what if you had been instrumental in reducing costs in the very area your boss was most concerned about? Does your boss know about this? You must dovetail your outcomes and the outcome of your partner. If you want to achieve truly excellent communication you must find a way to work with rather then against your partner's concerns.

But is it possible to get the outcome you want without knowing much about your partner? Can't you just convince them even though they don't understand or see a correlation between what they want and what you want? To do this is to *manipulate* and in doing so, you let yourself in for Genie LaBorde's "Four Dragons": resentment, recrimination, revenge and remorse.

All of the above is the foundation of excellent communication. However, when you actually sit face to face to talk, even if you know what positive outcome you want and you understand your partner's concerns, very little will occur unless you have rapport.

The word rapport is related to the French word *rapporter* which means to bring back. In English rapport means harmony, accord, or affinity. The French word is an action, the English a noun. Together they denote a process not just a state of mind.



Achieving rapport creates an understanding, an agreement between people. You feel a certain level of comfort in the situation. You know when you don't have rapport: your jokes fall flat–and so does everything else you try to communicate. How do you achieve rapport? LaBorde suggests the following.

First, to establish rapport you must trust the other individual's ability to accomplish the task. You must believe in their competence.

It is not necessary to *like* them but mutual confidence in competence for the task at hand is necessary. If you do not trust their competence, you had better deal with that directly. Here is a situation that explains this point:

A manager gives a report to an employee to write. However, the manager believes the employee's writing skills are not strong enough for the project. The manager is correct in that assumption. The manager hopes that this will be developmental for the employee. Of course, the employee is intensely frustrated by the line by line revisions that the manager has made in what is now the third revision. In anger the employee thinks, "Why didn't he write it himself if he wanted it all his way!" Much more than rapport has been lost. Now imagine the situation a different way.

The manager knows he will need someone on his staff with strong writing skills. So he takes an informal poll to see if anyone has a natural talent or inclination. After finding them the manager meets with them: "I have several projects coming up that will require very professional writing skills. All the work you do now is well written but these projects will require even more skill. Are you interested in developing even better writing skills? If so, I want you to find two or three writing courses that we can evaluate. I'd like to send you ..."

The outcome of the first approach is based on blind luck. The second approach has much more chance of success because the manager has dovetailed his own outcome with his employee's natural desire to progress.

Self-Esteem

As you move through the hierarchy of skills, learn to handle time and tasks, learn to think creatively, and learn to communicate excellently, you will find one elusive quality to be your greatest ally. It comes from trusting in your own abilities. It is called self-esteem.

Brian Tracy in his cassette series *The Psychology of Achievement* defines self-esteem as the belief system called "myself", and states that our level of self-esteem is determined by our answers to the following questions:

. Do I have the power to affect my life?

. Am I likeable? Do I like myself?

. Am I competent?

. Am I worthwhile?

How we answer these questions determines everything about our lives, including our careers. Cheering yourself on, taking gradual steps in the direction of change for the better, becoming vitally interested in your role as a manager, a director of change, will deliver you to a more satisfying work life and have many benefits outside work as well. Your ally is your own self-esteem and you can only get it from yourself.

Bibliography

Should any of the ideas presented in this paper interest you, the following is a list of resources–a very good place to begin.

de Bono, E. *Six Thinking Hats*. Boston: Little, Brown and Company, 1985

Kepner, C. and Tregoe, B. *The New Rational Manager*. Princeton: Princeton Research Press, 1981

Laborde, G. *Influencing With Integrity*. Palo Alto: Syntony Publishing, 1987

Lewis, D. and Greene, J. *Thinking Better*. New York: Rawson, Wade Publishers, 1982

Sinetar, M. *Do What You Love The Money Will Follow*. New York: Paulist Press, 1987

TAKE A DEEP BREATH

Margaret Brunner

Bay Area Resource for Cancer Control
1420 Harbor Bay Parkway, Suite 260
Alameda, California 94501-7080
(415) 748-6111

Look around. In almost every type of job, at almost every level of employment, you can see driven, stressed, alienated, unhappy people. Why? What causes emotional stress? Why should you be concerned about stress? What can be done about it?

In this paper, I will discuss the relationship between stress and disease as well as stress and productivity. The emphasis of this paper will be twofold:

1. What you as a manager can do to reduce stressors in your department.
2. What you as an individual can do to manage your own stress.

WHAT IS STRESS

Stress is a condition of the mind which reflects itself in the body. An event is thought of as stressful when it requires us to change or adapt. It makes no difference whether the event is a positive or negative one. The key to whether stress will cause harm to a person is that person's perception of their life circumstances. Has the stressful event exceeded their capacity to cope? A stress reaction, then, is the outcome of a person's subjective experience.

Subjective experience is the interaction of the five senses interpreted by the brain. This interpretation is relayed to the nervous system of the body.

THE PHYSIOLOGY OF STRESS

Millions of years of evolution have gone into developing the human nervous system. For most of this time, the reality of threats from a harsh environment demanded that the nervous system respond at a basic physical level.

Sophisticated, post-industrial western culture holds different threats for us. Usually these threats are not of a physical nature but of an emotional nature. However our bodies still respond to everything as if it were a physical

problem of survival. Physiologically, as soon as our nervous system perceives a threat, our bodies are instantly primed for the fight-or-flight response.

Physiological events occur in the body in response to stress via the combined actions of both branches of the autonomic nervous system. The first, or sympathetic branch, is primarily concerned with the body's signals traveling through the nervous system. This sets in motion a series of events enabling our bodies to respond quickly with intense muscular efforts through the release of epinephrine (adrenaline) and norepinephrine (noradrenaline). Intense biochemical reactions in the body produce a cascade of hormones, each one setting off the release of the other. The second, or parasympathetic, branch, keeps the body's maintenance systems in order and calms the body's responses to the sympathetic branch. These two mechanisms overlap and occur simultaneously. The known biochemistry of stress is quite complex. It currently includes more than 30 neurohormones and neurotransmitters.

A brief description of the internal changes that occur follows:

- Increased sugar and fats enter the bloodstream to provide more fuel for quick energy.
- Increased breathing rate increases the oxygen supply in the blood.
- Increased heart rate and blood pressure insure sufficient blood supply to the cells.
- Blood-clotting mechanisms are activated to protect against injury.
- Increased muscle tension prepares the body for action.
- Increased pituitary functioning stimulates endocrine production of the hormones; adrenaline and corticoid production increases. Digestive processes shut down; blood is diverted to muscles and the brain.
- Pupils are dilated, allowing more light to enter the eye.
- Attention and alertness increase.

But life these days requires that we frequently mask these physical responses to a threat. When your boss berates your performance, or a motorcycle cop flags you down, or a car cuts you off in rush hour traffic, your body is instantly mobilized by the threat. But physically fighting or physically fleeing the scene would be an inappropriate response. Therefore, you learn to suppress your reaction.

Throughout each day, you constantly override your body's response to your mind's perceived threat. When stress arousal is maintained over an extended period of time, the physiological systems begin to break down and undergo pathological changes. Your body parts wear out because of recurring hormonal imbalances. Your body will eventually break at the weakest link.

The human body is designed so that moments of stress do little harm IF the hormones released by the body are absorbed. But, unfortunately, the body does not automatically return to a non-aroused state or deactivate the released hormones. If the physiological response to stress is not discharged, there is a cumulative negative effect on the body. This is Chronic Stress or stress that is held in the body via unreleased or unused hormones.

EFFECTS OF STRESS

In 1988 the national budget for health care was $511 billion, or 11.4 percent of the gross national product. The average Fortune 500 company spends nearly a quarter of its after-tax profits on medical bills. Health insurance premiums are rising about 20 percent a year without any sign of stabilizing. It is estimated that between 60 and 90 percent of all visits to health care professionals are for stress-related disorders.

Stress leads to an incredible variety of disorders - from tantrums to heart attacks, from alcoholism to depression, from sleeplessness to colds. Some of the more serious illnesses to which stress contributes include hardening of the arteries, high blood pressure, heart failure, ulcers, diabetes, migraines, rheumatoid arthritis, ulcerative colitis, asthma, and cancer. Stress does not "cause" these illnesses; these illnesses are the result of the interaction of stress, inadequate coping skills, and pre-existing biologic conditions. Stress contributes to these illnesses in two fundamental ways. One way is by causing the body to physically deteriorate. The other way is by suppressing the immune system thus undermining the body's natural defenses against disease.

Of the people who will develop an illness that can be linked with stress, a percentage will either get heart disease or cancer. You might think that heart disease and especially cancer are diseases of old age and we all have to die of something. But these diseases strike many people in the most productive phase of their lives, including seemingly healthy men and women in their 50's, 40's, or younger. A quarter of all FATAL heart attack victims are under the age of 65. Forty percent of all cancer DEATHS occur in persons between the age of 25 and 64.

The impact of heart disease and cancer can be shown statistically as follow:

- At least 40 million Americans now have heart and blood vessel diseases.
- More than 600,000 Americans die each year of heart disease which is about half of all reported deaths.

- 5 million Americans alive today have had a history of cancer.
- In 1988, it was expected that more than 985,000 people would be diagnosed as having cancer of which 500,000 would die of it.
- Of every 10 deaths, two are from cancer and 5 from heart disease.

The following two sections describe the specifics of why this occurs.

STRESS AND HEART DISEASE

As you probably know, heart disease has primarily been viewed as a static accumulation of cholesterol. The discovery in the past decade of two new mechanisms has revolutionized the understanding of heart disease. These mechanisms can reduce or stop the flow of blood through the coronary arteries. They are:

- Coronary artery spasms, and
- Blood platelet clumping.

Heart disease is now seen as a dynamic process. As well as slow cholesterol buildup over time, sudden hormonal fluctuations in the body can cause either spasms or clumping to occur. The more cholesterol blockage there is in a coronary artery, the less spasm or clumping is required to completely close it off. Changes in coronary spasm and platelet function can increase or decrease the blood flow to the heart very rapidly. These mechanisms help explain why, for example, chest pain or a heart attack often occurs during times of emotional stress or immediately following a large, high-fat, high-cholesterol meal.

Prolonged and unabated stress response can lead to other heart conditions in the following ways. In chronic stress, the presence in the blood of corticoids (a hormone) increases. The corticoids can produce kidney disease through raising blood pressure. In this manner, a prolonged stress reaction can ultimately lead to severe hypertension due to kidney damage, which in turn aggravates the hypertension.

Pro-inflammatory corticoids can also cause tears in the arterial walls, which are then repaired by cholesterol plaques. Too much plaque is commonly called "hardening of the arteries", or arteriosclerosis. When this disease becomes advanced, it diminishes the supply of blood and oxygen to the heart. It is also not unusual for cholesterol plaques to become detached from the arterial walls and travel to the heart. Either of these two conditions can lead to coronary failure.

Normally it is the liver which monitors the amount of corticoids in the blood and which acts to reduce their levels when necessary. But during stress the liver's control mechanism is bypassed, permitting continued high concentrations of corticoids to circulate throughout the body.

It can be concluded that chronic stress profoundly effects heart disease via release of hormones into the blood stream and the by-passing of the body's normal channel of control.

STRESS AND CANCER

Cancer is a set of diseases characterized by uncontrolled abnormal cell growth. A current theory of cancer development proposes that all people produce abnormal cells in the body from time to time, either because of external factors or simply because of inaccurate cellular reproduction. Normally, the body's immune system keeps a close watch for abnormal cells and destroys them. For cancerous cells to grow, then, the immune system must be inhibited in some way. For the immune system to be inhibited, psychological change must be happening to the person to create this susceptibility.

External agents, radiation, genetics, diet - all four factors may play a role in the causation of certain types of cancer. None of them is a full explanation without considering why particular individuals, at particular points in their lives, contract cancer. If there is a genetic predisposition, it has been there all along. Their diet is likely to have been stable for a number of years. And, based on current medical theory, abnormal cells are present in everyone occasionally throughout life. At this point, I need to make clear the difference between abnormal cells and abnormal cells that become cancerous. All people have abnormal cells in their body. When abnormal cells are cancerous, it means that a mutation has occurred in the cell's DNA. This can occur from exposure to radiation, mutagenic substances, and cigarettes.

The lapse in the body's immune system that allows a life-threatening tumor to grow can be the result of stress. The research conducted by Dr. Holmes and Dr. Rahe at the Univ. of Washington School of Medicine has shown that major or accumulated stress predisposes people to illness. If a person feels threatened or incapable of resolving a situation, the probable outcome of this major or accumulated stress will be depression, despair, hopelessness, and helplessness. This state of emotional dejection and withdrawal is communicated to the hypothalamus which assists in controlling the immune system and plays a critical role in regulating the activity of the pituitary gland. This gland in turn regulates the remainder of the endocrine system with its vast range of

hormonal control functions throughout the body.

The immune system is designed in part to contain or destroy any cancerous cells. In this proposed model, the effects of stress produce a suppression of the immune system which leaves the body susceptible to the development of cancer. Let me make two distinctions about stress. Acute, short-term stress has no effect on cancer. An example of this is the sudden high level stress experienced having crashed your car into another one. Long-term acute stress, or chronic stress can both affect the growth of cancer. This is unreleased stress caused by long term situations over which the person feels helpless and hopeless. Immune suppression caused by these stresses can promote the more rapid growth of a pre-existing tumor allowing it to become clinically evident. Stress does not cause cell mutation which leads to tumors. Stress depresses the immune system which allows the growth of pre-existing tumors.

With this interaction of physiological and psychological events, optimal conditions are now created for cancer growth. At the same time that the body's defenses against intruders are lowest, the growth of abnormal cells is increased. The result can be cancer.

THE IMPACT OF STRESS IN THE WORK PLACE

Stress can be clearly seen as contributing to major diseases which claim more than 3 lives out of every 5. But more subtle costs than death are occurring daily, especially in our work places. This following section details some of the staggering costs of stress in terms of lost productivity.

It is estimated that 3 percent of American employees are absent every workday of the year. In wages paid, the cost of absenteeism is estimated to be $30 billion a year. The total loss, including lost productivity, is estimated to be as high as $150 billion. This lost productivity is the result of diminished functionality, disability, and absenteeism.

Recent research indicates that the majority of illnesses resulting in absenteeism are stress-related. In California, for example, stress disorders are the leading category of mental health disability claims. These claims increased 434 percent between 1982 and 1986.

The National Council on Compensation Insurance has indicated that compensation claims for stress-related disorders now account for more than 14 percent of all occupational claims, up from 5 percent in 1980. This national council warns that the growing number of stress-related worker compensation claims will soon strangle an already swollen liability system.

Three stress-related disorders - chronic pain, hypertension, and headache - are estimated to account for 54 percent of the absences, or $15.7 billion of the $30 billion lost in wages every year. Chronic pain, including upper and lower back and neck and shoulder pain, accounts for approximately 33 percent of the absenteeism. This means that chronic pain costs employers approximately $10 billion a year.

Hypertension is estimated to cause 11 percent of absenteeism and thus cost employers about $3.3 billion a year. Tension and migraine headaches are estimated to account for about 8 percent of absenteeism and thus cost employers about $2.4 billion a year.

As Dr. Frecknall, a psychologist involved with stress in the workplace, stated, *"It is clear that we are dealing with truly staggering amounts of money. If productivity is dependent on the physical and psychological presence of employees, if presence is dependent on health, if health is affected by stress, and if stress can be managed, then a fundamental change in employees' basic life assumptions could have an enormous effect on productivity."*

STRESS MANAGEMENT

While stress will predispose a person to illness, the significant factor still is how a person copes with the situation. Not everyone responds to the same pressure and not everyone is affected equally by it. Just as some people seem to naturally have sturdy physical constitutions, others seem to be psychologically resilient. Somehow they manage to ride through the major crises of life, while others crash and burn.

Dr. Kenneth Pelletier, a leading researcher in the mind/body field, has identified 3 major characteristics of those who are "stress-immune"; that is, those who can successfully cope with stressful events. These 3 characteristics all deal with the attitudes or beliefs that a person has towards life. These coping skills are:

1. They perceive stressful situations as challenges or opportunities, not as problems.
2. They commit themselves to the situation fully. They take a stand and act upon it. They experience an internal locus of control (which indicates a person's sense of being effective in the world rather than a passive victim of circumstances).
3. They have goals, values, and beliefs by which they live their lives. Their goals and actions are based upon their personal ethics.

Stress is a reaction to a situation "perceived" as a

threat. If a person's orientation to life allows them to see situations as challenges or opportunities and their approach to these challenges is to participate fully, then stressful events will not affect them harmfully.

William H. Foege, Assistant Secretary of the U.S. Department of Health and Human Services, writing in the Journal of the American Medical Association, expressed his concerns this way: *"The scientific basis for the influence of lifestyle choices on health continues to grow... In the coming decades, the most important determinant of health and longevity will be the personal choices made by each individual. This is frightening for those who wish to avoid such responsibility and exciting for those who desire some control over their own destiny."*

Pelletier states that the question for modern man has become: Am I going to live a stressful lifestyle and seek traditional, disease-responsive medical help for my stress-related problems, or am I going to take responsibility for creating a more healthful lifestyle that will enable me to avoid many of those disorders? We will all answer this question one way or another.

The approach to stress management can be summarized as a three-fold approach:

- You can remove the stressors from your environment or avoid them.
- You can change your outlook on life to a "stress-immune" one.
- You can learn techniques that will return homeostasis (hormonal and electrical balance) to your body.

The following suggestions are inexpensive, require no special equipment or advanced technology and are without dangerous side effects. In these respects, they are the opposite of many current medical practices. They are based on the premise that changing what seem to be the causes of leading diseases is a better approach than just treating the disease.

REMOVING STRESSORS FROM YOUR WORK ENVIRONMENT AND THAT OF YOUR EMPLOYEES

1. Make challenge level and skill level closely matched and high.
2. Give clear feedback. Make sure there is an objective way to evaluate progress. Make your expectations very clear.
3. Allow staff to control their work. Do not give responsibility without the necessary authority to follow through.
4. Provide work in which the motivation is intrinsic. The action itself is rewarding.
5. Allow people to set clear-cut goals for themselves.

6. Give praise publicly.
7. Thank people for their performance as close as possible to the time of the performance. Do it in person, and if possible, in writing.
8. Be consistent.
9. Listen twice as much as you talk. Make it easy for people to talk to you. Make confidentiality a rule.
10. Follow through on what you say you will do.
11. Give criticism privately.
12. Never criticize the person, only the action or the product. Give the criticism as close as possible to the time of the inappropriate action.
13. Give clear, complete instructions.
14. Set mutually agreed upon deadlines.
15. Don't be a dictator. Allow a person leeway in how a task will be done and when it will be done within agreed upon parameters.
16. Offer encouragement, not ridicule. Being too critical is counterproductive. Accept the fact that errors and mistakes are bound to occur. No one is going to continue doing something if you belittle their attempts.
17. Make work fun. Encourage social networks to occur. People who like being part of your team are more likely to show up for work, and contribute positively to the effort.
18. Never hire someone you would be unwilling to go out to lunch with. If you are unwilling to go out to lunch with them, you will most likely be unwilling to spend much time with them explaining the work, the environment, your requirements, etc.
19. Rotate unpopular jobs.
20. Provide the necessary tools and the proper physical environment for the job (well lit, well ventilated, spacious and quiet).

TECHNIQUES TO CHANGE YOUR OUTLOOK ON LIFE

If you are interested in controlling your stress level, you can opt to make either major changes or minor ones. Many articles have been written on the minor changes such as getting a housekeeper, therefore, I chose not to detail them. My focus is on significant major changes which can profoundly effect your life. The following list of lifestyle choices truly makes such an impact. But remember, major change can be traumatic. If you are considering implementing them, do so gently and ease yourself into a lifestyle change gradually. Do NOT do all of them at the same time. Introduce change in simple, incremental steps that are congruent with your present life and concerns.

1. Do things that bring you a sense of fulfillment, joy, and purpose. See your life as your own creation, and strive to make it a positive one.

2. Pay close and loving attention to yourself, tuning into all your needs. Take care of yourself, nourishing, supporting, and encouraging yourself. Know your values and your beliefs. Use them as your reference point when making decisions.
3. Release all negative emotions - resentment, envy, fear, sadness, anger. Express your feelings; don't hold on to them.
4. Hold positive images and goals in your mind; pictures of what you truly want in life. When fearful images arise, refocus on images that evoke feelings of peace or joy.
5. Love yourself, and love everyone else. Make loving the purpose and primary expression in your life.
6. Create fun, loving, honest relationships, allowing for the expression and fulfillment of needs for intimacy and security. Try to heal wounds in past relationships, as with old lovers, and mother and father.
7. Make a positive contribution to your community through work or service that you value and enjoy.
8. Make a commitment to health and well-being, and develop a belief in the possibility of total well-being. Develop your own program for well-being, drawing on the support and advice of experts without becoming enslaved to them.
9. Accept yourself and everything in your life as an opportunity for growth and learning. Be grateful. When you screw up, forgive yourself; learn what you can from experience, and then move on.
10. Think creatively. Change negative self-talk to positive. Take responsibility for your inner world. Choose loving words over hostile ones, peaceful thoughts over disturbing ones. Observe yourself in a spirit of curiosity, letting go of self-judgment.
11. Set realistic goals for yourself. Edit your "TO DO" lists down to what is essential and possible. Make some short enough you can finish them. Find ways to reward yourself for each goal reached before rushing on.
12. Slow down; enjoy life. Take the scenic route. Choose more time over more money.
13. Trust in and accept the present. Once you see and accept what is, it can change.
14. Communicate honestly with yourself and others. See yourself and others as being "on the same side." Tell what's true for you, directly and simply. Honesty may mean saying NO or being told NO. This is natural and need not detract from you feeling loved. Ask for what you want and encourage others to do the same. Hug the ones you love.
15. Be generous with yourself and others. Forgive readily, avoid comparisons, and release judgments. Accept yourself just as you are. Be gentle and compassionate with yourself. Everyone will benefit. Lasting change follows self-acceptance. Set realistic expectations for yourself and others. Aim for excellence, not perfection.
16. Keep a sense of humor.

RETURNING HOMEOSTASIS TO YOUR BODY

Pelletier has expressed that the essence of stress is disharmony. In bodily terms, it is a difference in electrical activity in the two hemispheres of the brain. To bring homeostasis back to the body, the brain must shift from asynchronous activity to synchronous. Four activities can restore harmony or make the brain's electrical activity synchronous. These are:

1. Humor as in laughing with your whole being.
2. Problem solving at the point where the "ah ha" occurs.
3. Deep meditation.
4. Orgasm.

As he says, *"Its not a bad list to choose from."*

The only one of the four that I have chosen to explain in some detail is meditation. I have included deep breathing techniques as introductory exercises to deep relaxation and meditation in Appendix A.

Relaxation/meditation techniques are any activities that keep the attention anchored in the present moment. These techniques are explained in detail in Appendix B. They are experiential exercises requiring a person's whole attention. These relaxation/meditation activities are geared to allow your mind to let go of all thoughts pulling you out of the moment. In the relaxation or meditation activity, the mind takes a back seat to just being. It is this meditative state that elicits the body's relaxation response of hormonal balance and inner peace.

Numerous research studies have demonstrated that meditation is psychologically and physiologically more refreshing and energy restoring than deep sleep. Psychophysiological states achieved during successful meditation are quite markedly different from those of both normal waking consciousness and deep sleep. The physiological changes carry over into a person's daily life. Meditators have been found to be more psychologically stable, more autonomically stable, less anxious, and to experience an internal locus of control.

One of the reasons why meditation produces this carry over effect is that it helps an individual learn to maintain low arousal states of neurophysiological functioning. This minimizes stress reactivity in a stressful situation. Relaxation is the physiological opposite of tension and anxiety. When a relaxed state is induced in the presence of a stressful threatening event, the physiological reaction to stress is inhibited.

Concentration is essential in all systems of relaxation/ meditation. The meditator learns to fix his attention firmly upon a given task for increasingly protracted periods of time. This concentration of attention overcomes the mind's usual habit of flitting from one subject to another. When the incessant activity of the mind is stilled, the meditator experiences that aspect of his being which is prior to and distinct from his thoughts and from attention itself.

Meditation is a discipline. It must be worked with consistently in an organized manner in order to have any meaningful effect. Doing a particular exercise once does not have any value other than to give you an experience of it. The same is true of meditation.

IN CONCLUSION

As you have no doubt concluded, stress is part of life. How you chose to deal with it determines the quality of your life.

My favorite way to think about stress and stress control techniques is with the metaphor of the surfer. I see myself as being a surfer and I see life as being a beach.

On a beach there is sand and crashing surf. There is no way to control the surf. No one even tries. What a surfer does is learn to ride the waves gracefully without wiping out. The surfer lives intensely in the moment and has a single focus.

If he or she wipes out, then they retrieve their surfboard and jump in again. Between waves, they paddle gently in the ocean while eyeing the next wave. Periodically they retreat to the beach to fully relax. They enjoy themselves and surf with a passion.

The surfer represents the skillful negotiation of the pounding surf of life. Like the surfer, ride the highs and lows with equal balance. Make constant adjustments to go with the wave and yet retain control. Surf through and beyond the swells of anger and waves of fear. Discriminate among emotional waves, knowing which to ride and which to let pass. Avoid the stagnation of the low tides. Rest and recuperate between waves.

APPENDIX A - DEEP BREATHING TECHNIQUES FOR RELAXATION

DEEP BREATHING.

1. Breathe in slowly through your nose as you expand your belly. Imagine that you have a balloon in your belly and, as you inhale, you are slowly inflating it, causing the abdominal area to swell.
2. Breathe out slowly through your nose. Pull your abdominal muscles in as you press all the air out of your lungs.
3. Continue breathing in and out as you abdomen rises and falls. Establish a natural rhythm.

This breathing pattern should be used all the time. This is the way babies breathe. It is the natural way to breathe to counteract stress.

THE CLEANSING BREATH.

This is a good, quick exercise you can practice anytime you want to release tension.

1. Inhale deeply through your nose.
2. Exhale through your puckered mouth, as if you were blowing out a candle.
3. Repeat steps 1 and 2 three times.
4. Issue a few sighs. Inhale deeply, then sigh. Again. Again. With each sigh, drop your chin to your chest and droop your shoulders. Think of yourself as a tire letting all its air out. Think of the tension you are releasing.

THE STRESS-DISCHARGING BREATH.

Allow more time to do this exercise. It's perfect for after work.

1. Start relaxing with several abdominal breaths; breathe in to the count of four, then breathe out to the count of eight.
2. Take a deep breath through your nose and hold it. Tense your feet as long as you can. (WARNING: If you have a history of heart disease, high blood pressure, or stroke, consult your physician about this technique. They may suggest a modified version, with little or no breath-holding.)
3. Relax your feet as you exhale with a sigh through your mouth.
4. Take a few abdominal breaths with each count, as in step 1.
5. Breath in deeply through your nose. Hold it. Tense your

calves.

6. Relax your calves as you breath out with a strong exhalation.
7. Repeat the sequence for each part of your body, working from the extremities to the center: feet, calves, thighs, buttocks, abdomen, fingers, forearms, upper arms, shoulders. (Hunch your shoulders up to your ears). Don't forget your face - work it in three stages: Pull your jaw back so your mouth looks funny; scrunch up your nose; furrow your brow.
8. Take a few minutes to relax and let go.

APPENDIX B - RELAXATION/MEDITATION TECHNIQUES

During relaxation the body may discharge tension through tingling, involuntary jerking, etc. Do nothing, this is normal. This is a physiologically non-damaging release of neuronal impulses. They are considered to be normal responses to the body's attempt to self-regulate toward a state of increased balance.

These relaxation techniques produce physiological changes. If you have high or low blood pressure, diabetes, hypoglycemic conditions or heart conditions, you should consult with your physician before you even start. Those with mental or emotional disorders should consult with their therapist before starting. Massive motor discharges may be quite pronounced in those having a history of alcoholism, severe accidents, or incipient epileptic behavior. Crying spells may be associated with motor discharges in those with a history of traumatized birth, abortion, traumatic dental surgery, or suppressed emotions over a significant incident such as the death of a relative. If you find yourself suffering disquieting side-effects, then practice only under the supervision of trained instructors in these techniques.

General Directions:

1. Choose a quiet spot where you will not be disturbed by other people or by the telephone. Unplug the telephone if necessary.
2. Dim the lights if possible.
3. Wear loose clothing, or make your clothing loose by unbuttoning your belt, loosening your tie, taking off your shoes, etc.
4. Sit on the floor or a comfortable chair or couch.
5. Sit in a relaxed position with your spine straight and shoulders back (but down and loose) to allow your lungs to expand.
6. Rest your hands on the arms of your chair or in your lap.
7. If you are in a chair, place your feet flat on the floor.
8. Keep your body in an open posture - that is, do not cross your legs or cross your arms in a way that can cut off your circulation.
9. Close your eyes to make it easier to concentrate.
10. Relax your muscles sequentially from head to feet. This step helps break the connection between stressful thoughts and a tense body. Just become aware of your body parts in succession, letting go as much as you can as you breathe. Cooperate with gravity and let go of tenseness in your body. Starting with your forehead, become of aware of tension and let it go. Go through the rest of your body in this way proceeding down. This need only take a minute or two.
11. Don't worry about how you are doing. As soon as you worry about whether you are doing it right, you have

shifted to anxiety. If you notice yourself doing this, try labeling it as "judging", then let go, coming back to the breath and the focus of the relaxation technique or meditation.

12. Your mind will not stop for more than a second or two, if at all, so don't expect it to. What happens is that the part of yourself that watches the mind is learning to "use its muscles". Each time you notice you have drifted into thought, try labeling where you were; for example, "judging", "thinking", "planning", and then let it go, getting back to the exercise. The most common complaint about meditation is "I can't stop my mind from wandering." Don't try. Just practice bringing it back to focus and concentrate on the breath whenever you notice it wandering.

RELAXATION TECHNIQUE

AUTOGENIC TRAINING.

In a comfortable position, slowly recite the following instructions. Breathe deeply and evenly; and recite the verbal cues to yourself as you exhale.

1. "My hands and arms are heavy and warm" (5 times).
2. "My feet and legs are heavy and warm" (5 times).
3. "My belly is warm and comfortable" (5 times). (Omit this step if you have ulcers.)
4. "My breathing is deep and even" (10 times).
5. "My heartbeat is calm and regular" (10 times). (If you experience a rapid, irregular heartbeat - inset the word "slightly" in front of "calm" when repeating this phrase).
6. "My forehead is cool" (5 times).
7. "When I open my eyes, I will remain relaxed and refreshed" (3 times).

Then perform the following sequence of body movements.

1. Move your hands and arms gently.
2. Move your feet and legs gently.
3. Rotate your head.
4. Open your eyes and sit up.

Instead of memorizing the verbal cues, you might try to record them into a tape recorder and listen to the tape. As you listen, silently repeat the instructions to yourself as you do them.

MEDITATION TECHNIQUES

Meditation is concentrating on one thing at a time. The meditator's task is to discipline his mind to be as alert as possible with just one thought being the entire focus of the mind.

Dr. Lawrence LeShan, a prominent authority on meditation, commented that to say meditation is very difficult is to understate the case. He states that you will realize that if your body were one tenth as unresponsive to your will as is your mind, you would never get down a flight of stairs without half-a-dozen broken bones. I agree.

There is no right way for everyone to meditate. Within the general discipline, what is the best way for one person may well be the hardest for another.

THE MEDITATION OF BREATH COUNTING.

In this meditation, the object again is to be doing just one thing as completely and fully as possible. In this case the one thing is counting the exhalations of your breath (your breathing out). You strive to be aware of just your counting and to be as fully aware of it as possible. All your attention is gently and firmly and repeatedly brought to bear on this activity. The goal is to have your whole being involved in the counting.

In this exercise one is paying as full and complete attention as possible to the counting itself. Thoughts, feelings, impressions, sensory perceptions, to the degree that they are conscious, are a wandering away from the instructions. The exercise involves counting up to 4 exhalations of breath and then repeating.

- Breathe in, count "one" on the outbreath.
- Breathe in, count "two" on the outbreath.
- Breathe in, count "three" on the outbreath.
- Breathe in, count "four" on the outbreath.
- Repeat the process.

When you find yourself thinking about your counting (or about anything else), you are wandering away from the instructions and you should bring yourself gently back. If you find yourself modifying your breathing, this also is a straying from the exercise.

Start with 15 minutes at a time on a daily basis. After a few weeks, increase to 20 minutes, and after a month to 30 minutes. After working at this last schedule for a month or two you should be able to determine your own future course with this meditation.

THE MEDITATION OF THE BUBBLE.

On meditations of this sort, you observe your own consciousness in a special way, while interfering with it as little as possible. You meditate on the stream of your own consciousness.

Picture yourself sitting quietly and comfortably on the bottom of a clear lake. Picture large bubbles rising slowly through the water. Each thought, feeling, perception, etc., is pictured as a bubble rising into the space you can observe, passing through and out of this space. It takes five to seven or eight seconds to complete this process. When you have a thought or feeling, you simply observe it for this time period until it passes out of your visual space. Then you wait for the next one and observe it for the same amount of time, and so on. You do not explore, follow up or associate to a bubble, just observe it with the background of " oh, that's what I'm thinking or feeling or sensing now. How interesting." Then as it passes out of your visual space (as the imaginary bubble rises), you calmly wait for the next bubble.

Do not be thrown off the meditation if the same "bubble" rises several times. If you just go on, this will pass. And do not be disturbed if you cannot see the connection between the bubbles or the source of your thoughts. If you simply stay with the discipline long enough, most confusing connections will clear up. If your mind seems to go "blank," why, feeling "blank" makes a fine bubble!

The purpose of the concept of bubbles rising through the water is to help you do two things. The first is to keep the timing. You learn to simply contemplate each thought or perception for approximately a definite time and then let it go. Secondly the structure helps you look at each one individually and not constantly feel you must make connections between them.

Since there are crucial reasons for this structure, those who find the idea of sitting at the bottom of a lake unsympathetic or disturbing can picture themselves on a warm, windless day on the prairie watching large, separate puffs of smoke rise from a campfire as if it were an Indian signal fire.

Start with 10 minutes a day for two weeks. If you have difficulty with one after several days, try the other concept. After two weeks, go to 20 minutes a day or 30 minutes if it is clearly the right meditation for you. Do this for 3 weeks to a month. At the end of that time you should know how to include this meditation in your own program.

REFERENCES

American Cancer Society, "Cancer: Basic Data" in Cancer Facts & Figures - 1988.

Borysenko, Joan, Ph.D., "Getting Back in Control", in New Age Journal (May/June 1987).

Borysenko, Joan, Ph.D. with Larry Rothstein *Minding the Body, Mending the Mind* (Massachusetts: Addison-Wesley Publishing Company, Inc., 1987).

Csiksyntmihalyi, "Stress, boredom caused gifted students to lose interest in their special fields", in Brian/Mind Bulletin (September 1988, Volume 13, Number 12).

Faelton, Sharon; David Diamond, and the Editors of Prevention Magazine *Take Control of Your Life, A Complete Guide to Stress Relief* (Pennsylvania: Rodale Press, Inc., 1988).

Frecknall, Peter, Ph.D., "How Much Can Mind-Body Techniques Save by Reducing Absenteeism?", in The Journal of Mind-Body Health Advances (Spring 1989 Vol. 6, No. 1).

Fulghum, Robert *It Was On Fire When I Lay Down On It* (New York: Villard Books, 1989).

Hall, Stephen S., "A Molecular Code Links Emotions, Mind, and Heath", in Smithsonian (June 1989).

LeShan, Lawrence, Ph.D., "Alternate Realities", in Magical Blend (Issue Eleven).

Levey, Joel, *The Fine Art of Relaxation, Concentration, and Meditation, Ancient Skills for Modern Minds* (London: Wisdom Publications, 1987).

Locke, Steven, M.D., and Douglas Colligan *The Healer Within, The New Medicine of Mind and Body* (New York: E.P. Dalton, 1986).

Nuernberger, Phil, Ph.D. *Freedom From Stress, A Holistic Approach* (Pennsylvania: The Himalayan International Institute of Yoga Science and Philosophy of the U.S.A., 1981).

Ornish, Dean, M.D. *Stress, Diet, and Your Heart* (New York: Holt, Rinehart, 1982).

Pelletier, Kenneth R., Ph.D. *Mind As Healer, Mind As Slayer, A Holistic Approach to Preventing Stress Disorders* (New York: Delacorte, 1977).

Pelletier, Kenneth R., Ph.D. "Optimal Health", presentation to the Institute for the Advancement of Health 1989 San Francisco Speaker Series, taped transcript.

Pelletier, Kenneth R, Ph.D. and Peter Frecknall, Ph.D.,

"Mind-Body Health: For Workers, Employers, and the Nation", in The Journal of Mind-Body Health Advances (Spring 1989 Vol. 6, No. 1).

Pelletier, Kenneth R, Ph.D. and Robert Lutz, Ph.D., "Mind-Body Goes To Work", in The Journal of Mind-Body Health Advances (Spring 1989 Vol. 6, No. 1).

Perkins, John M., *The Stress Free Habit, Powerful Techniques for Health and Longevity from the Andes, Yucatan, and Far East* (Vermont: Healing Arts Press, 1989).

Prather, Hugh *Notes on How to Live in the World...And Still Be Happy* (New York: Doubleday & Company, Inc., 1986).

Siegel, Bernie S., M.D. *Peace, Love, & Healing, Bodymind Communication and the Path to Self-Healing: An Exploration* (New York: Harper & Row, 1989).

Siegel, Bernie S., M.D., "The Psychology of Illness and the Art of Healing", a presentation to the 1989 Association for Humanistic Psychology 27th Annual Conference, taped transcript.

Simonton, O. Carl, M.D.; Stephanie Matthews-Simonton, and James L. Creighton *Getting Well Again* (New York: Bantam Books, 1978).

Williams, Redford, "The Trusting Heart", in Psychology Today (January/February 1989).

The Do-It-Yourself Career Advancement Kit

by David M. Scott, CPIM
Manager, Computer Integrated Manufacturing
AMP Packaging Systems, Inc.
700-E Jeffrey Way
Round Rock, TX 78664
(512)244-5100

INTRODUCTION: THE THREE STEPS TO MOVING UP

Why "Do It Yourself"? Because no one else will do it for you. No matter what field, industry or position you're in, it's your responsibility to decide where you want to go in your career. The Horatio Alger myth tells us that virtue is its own reward; if you work hard and keep your nose clean, merit will be recognized and you'll rise to the top. Perhaps. However, your chances of rising are better if you pay attention to managing your career.

The first step in moving to management is QUALIFICATION. You must prove that you have the skills and abilities required for the job. Because managers and non-managers do different jobs, it is not enough to do your own job well. You have to identify the skills required of a manager and develop them in yourself.

Business organizations are miniature models of society, and they follow the same patterns seen at the macro level. To become a member of the next level up you must go through the SOCIALIZATION process. To be a manager, you have to be seen by other managers as "management material". Part of this image comes from the quality of your work. Managers are expected to be above the average level of competence. The other part comes from observing the social norms and rituals of the management subculture.

Finally, those with the power to promote you must be made aware of your abilities and your desire to move into management. This is the FAMILIARIZATION process. Its purpose is to "sell" you as a manager, without making you look self-aggrandizing or scheming.

WHAT IS A MANAGER, AND DO YOU WANT TO BE ONE?

If we are going to discuss becoming a manager, we need to define our terms. The word "Manager" in your title does not necessarily make you a manager; there are a lot of "System Managers" out there who are really senior technicians. To be a manager, the following conditions must exist:

1. You are held responsible by the company for the actions of those who report to you.

2. You make decisions concerning the use of company resources.

3. You supervise people. This includes assigning work, evaluating performance, and intervening when necessary.

The essentials of management are RESPONSIBILITY, AUTHORITY, and SUPERVISION. Managers do other things besides these three, but these are the things that distinguish a manager from others.

This paper assumes that the reader wants to advance to management levels. This is not true of everyone. A case in point is that of Sandra, the most experienced and technically competent programmer analyst in a large manufacturing company. During a reorganization, management recognized her years of service and her unparalleled talents by making her Programming Manager. After eight unhappy months, she asked to be removed from that position and reassigned to a technical job. Why? Because, in her new position, she was no longer doing what she did best, which was programming. Instead, she was performing the functions of a departmental manager, which are quite different.

The first decision you have to make, then, is whether you want to move into management or stay on a technical career path. There's nothing wrong with either decision, and there's a lot to be said for choosing the path that suits you best. You should also make this decision before you're offered a promotion into one path or the other, in order to avoid the problem that Sandra had.

Obviously, the thing to do is to be prepared for that next job up, by determining what talents are needed and developing them yourself. The "Do-It-Yourself Career Advancement Kit" outlines a basic set of management specific skills to get you started.

STEP ONE: QUALIFICATION

This section is an inventory of critical skills that managers require. By acquiring these skills, you can make yourself more promotable. By themselves, however, these skills do not guarantee promotion. The following sections of this paper discuss ways to manage your career to maximize your chances for advancement.

This brief discussion can't teach you all of these skills. We'll hit the high points; it's up to you to find the educational resources and study the subjects in depth.

Financial Skills:

Once a manager, you'll plan and control a budget. In addition to budgeting dollars, you may also have to budget manpower, floor space, machine time, etc. Every company has its own procedure for budgeting, but there are some general rules:

1. Start with the company's business plan. You have to know where you're going before you can plan to get there. Look for trends that affect your needs for manpower, equipment, supplies, and facilities. Know which aspects of your workload vary with changes in levels of sales or production.

2. Look at past actual to budget performance. Find out how the previous budgets were derived, and see how accurate or inaccurate they were. Remember that those who do not understand history are condemned to repeat it.

3. Your budget is not a Christmas list. Sound out management on the feasibility of large expenditures. Turning in an unrealistic budget can damage your credibility, especially if you try to defend the indefensible. DP people are sometimes accused of buying technology for its own sake. It's up to you to destroy this stereotype.

4. Once you've budgeted, keep track of how you're doing. Your first indication of a variance from budget should not be your boss storming into your office.

A sort of corollary to budgeting is capital authorization. When you propose to make a large expenditure, you'll usually have to justify it with numbers. Things to look for are:

1. What are the costs of working with vs. without the item? Include manpower, time, materials, and even electricity. Several companies justified replacing their old HP7933 disc drives with the new "Eagle" series mainly on the savings in power draw and air conditioning BTU's in the computer room.

2. What benefits will you get from the new item? By "you", I mean the company, and above all the customer. The three basic items any business must deliver to the customer are quality products, reliable delivery and competitive price. If you can show improvements in one of these areas, it will go a long way toward justifying the purchase.

3. What is the Return on Investment? How much dollar benefit will the company get out of this purchase, as a percentage of the purchase price? This percentage has to be higher than you can get by putting the money into something else. If the company can buy bonds that pay 10%, your project had better pay more, or it'll have a hard time getting approved.

4. Along with ROI goes the Payback Period. Given that there's a certain dollar value savings in using your proposed purchase, how many months worth of those savings will it take to break even on the cost of the purchase?

Personnel Skills

As a manager you'll have to staff your department. This involves interviewing people, choosing the one to hire, and breaking them in to your way of doing things. But your responsibilities don't stop there. You'll have to do periodic performance evaluations. You may conduct training. And, the biggest problems for some managers: exercising discipline and terminating an employee who's not up to standard. Many skills required in the personnel area can't be learned in a classroom; they come from experience. If you don't have that experience, consult someone who does -- your supervisor, your company's Personnel Manager, or an experienced manager with whom you're comfortable. Rules to remember:

1. Think before you speak or act! Often, it's tempting to lash out with the sword of authority when you find an employee doing something seriously wrong. But once you speak, you can't recall the words. Blowing your stack at an employee is unprofessional, and can get you into a lot of trouble. Take time out, calm down, and make a clearheaded evaluation of the situation. If you think disciplinary action is called for, discuss it first with your Personnel Manager. Make sure that the action you plan isn't a violation of labor laws, company policy or union agreements. Once you've arrived at a course of action, see the next item.

2. Never discipline an employee in public. Humiliating someone in front of peers is unnecessarily cruel, and if the person has a sympathetic audience, he (or she) may try to argue the matter, counting on group pressure to make you back off.

3. Be Prepared. When it's time to do an employee evaluation, don't try to remember everything the employee's done for the last twelve months. Keep records for each employee, to include work assignments, attendance, training taken, personal achievements (night school courses, professional certifications, etc.), and disciplinary actions. This will help you not only in doing reviews, but in assigning future projects, writing job descriptions, and filling out a personnel requisition for a new hire. The record can also be useful in backing up your decisions on discipline, or if you're told by management to select someone for a layoff or staff cut. Not pleasant, but facts of corporate life.

   Also along the "be prepared" lines, the way to conduct a successful interview is to prepare in advance. If you're interviewing, go over each candidate's resume with a copy of the job description in front of you. List the questions you want to ask the candidate, and information you want to give. Because the first few minutes of an interview are the hardest for both parties, you may want to write out your "opening speech", and practice it beforehand.

4. Learn from those who have experience. If you're not sure how to handle a situation, talk it over with your boss. Despite what a lot of people say, bosses are human. They don't expect you to know everything the day you take over your new job. It's OK to ask for help.

Operational Skills

This class of skills can be described as the skills needed to work like a manager. You'll see what I mean as you read on:

1. Project management involves planning toward a defined goal, allocating people and resources, monitoring progress, making midstream corrections, and assessing performance both along the way and when the project's done. You should know what a PERT chart and a Gantt chart are, and how to read each one; be familiar with the terms critical path, bottleneck, deliverable, and resource leveling.

2. Presentation skills are a powerful weapon in the manager's arsenal. Some excellent proposals have died on the vine because they were poorly presented, and some real "dogs" of projects have gotten approved because somebody sold them well. Make a point of developing your oral and written presentation skills. Practice makes perfect in this area. A word of warning: never use sophisticated presentations to get your pet "dog" project approved. Sooner or later, the boss will discover its lack of true merit.

3. Negotiating is a skill that few people develop to its full potential. Most think of negotiating as an adversary situation, where one side wins, one loses. It is possible, and even common, to have a win-win situation. This may be unfamiliar territory to those who played competitive sports in high school or college. Many successful sales people are former athletes; they've been culturally conditioned not to accept the losing end of a situation. But negotiating is not selling; it's exploring alternatives,

listening to the other guy, and working together to get as close to best-case as possible for both sides.

4. Teaching skills are a special case of presentations. Coming from DP, you have specialized knowledge that needs to be distilled into non-computerese and transmitted to others. Some DP people lose patience with fellow-workers who are not "computer literate". This accomplishes nothing, except to alienate the people you're trying to help. The best teachers don't talk all the time; they listen a lot, and sometimes they ask questions not to find out what the student knows, but what they don't know. This is another area that improves with practice.

5. Know your company. No matter what kind of company you're in, you must have a general understanding of what the company does and how it does it. Talk to some of the people directly responsible for the product or service your company provides. If you work for a manufacturing firm, get out on the shop floor and see how the product goes together. In a marketing firm, sit in on a presentation to a client.

From this section, you can see the difference between the kind of skills a programmer needs and the skills required in management. Many people with technical backgrounds are uncomfortable with these areas, because they are much less well defined than, for instance, COBOL syntax. Acquiring these skills requires a variety of approaches. Business school courses and seminars can help with financial and budgeting skills; you can practice project management, training and presentation skills on your own projects; you can volunteer for leadership positions in a user group or professional organization to get a feel for being in charge. Work on developing interpersonal skills for negotiating and personnel management whenever you interact with others.

STEP TWO: SOCIALIZATION

Business organizations are, as mentioned before, miniature models of society. If you want to fit into any particular group within a society, you have to understand that group's values and know how to participate in its approved behaviors. Think of it as moving to a new school, an experience most of us had as children. The object is to "fit in"; to become acceptable to the group.

This is a problem area for some people. They believe they lose part of themselves in the process, and somehow "sell out" by adjusting to the group values of management. If this bothers you, perhaps you should reconsider your career objectives. An extreme example: Karl Marx was not "management material". Why? Not because he couldn't organize - he organized a whole political movement. Not from lack of leadership ability - he had millions of followers, and still does. Not from lack of intelligence or familiarity with the economy - he was a very competent political economist. He expressed himself well orally and on paper. Marx had the basic skills and characteristics of a good manager; what he lacked was the shared values of management.

The process of socialization depends on three basic factors:

1. A common language.
2. Shared values.
3. Social norms and rituals.

Identifying these in your own company takes shrewd observation. The company "society" has subgroups divided along lines of department, occupational skill, common hobbies, etc., as well as the more basic lines of race, sex, religion and age. These groups overlap, and at times, people will act differently according to the group with which they identify in the situation. Your task is to identify the members of the group to which you aspire, observe them when they act their roles as group members, and discover the group's language, values and behaviors.

The Common Language

People tend to use one "dialect" within their own department, and another talking to other departments. When people from different subcultures (departments) talk, they tend to look for and use a common "dialect" to make understanding easier. Because managers spend a large portion of their time talking across department boundaries, you should analyze manager-to-manager talk in this context. How do you analyze it?

1. Look for mutually understood terms. If one manager refers to "making this month's numbers", the other will understand which numbers are referred to, even if you do not. A little questioning will bring out the true meaning of "making the numbers", whether it be hitting a sales forecast, holding down product rejects, etc.

2. Look for "borrowed" dialects. If the managers in your company say things like:

   "I think we can do an end run around the Finance Committee on this project"

   "Did you see Marketing's third quarter numbers? They're really going for the long bomb!"

   "Joe's calling the signals on that project. They're not making any yardage at all."

   You can assume that the managers borrow football terms, and that they will understand you if you refer to a critical approaching deadline as a "fourth and inches" situation.

3. Use context to filter out language of other subgroups. Just because two managers are talking, it does not mean that they're speaking "management". They may both be avid golfers (a common subgroup), and their use of terms like "rough lie", even if in reference to work, may not be part of the management dialect. Don't jump in and try to use this terminology unless you too are a golfer.

Once you have learned the management dialect at your company, you can begin to employ it in conversations with management. Go easy at first, and don't make the mistake of using it on your peers who are not managers. You'll be speaking a dialect unfamiliar to their subculture, and you'll mark yourself as an outsider.

Shared Values

The idea behind shared values is "what's really important to management?". This is sometimes hard to determine. Just like individuals, management as a group will have two sets of values: one one published, the other implied. You will get the first set by asking, but it is the second set that really matters.

The published values of a management group, or any group, are usually very worthy ideals. They are often expressed by such statements as"

"The customer comes first."

"People are our most important asset."

"Product quality is number one."

These are all excellent sentiments. To find out if they are the true values in force, look for the following:

1. How much effort goes toward these goals? If Customer Service can't get another service rep hired because of budget constraints, then maybe conformance to approved budgets is more important than customer service.

2. What gets measured in your company? If sales booking and billing figures are published daily to the whole company, and quality statistics are collected and stored only by the Quality Control department, you can make some assumption about the relative importance of sales and quality.

3. How do managers express a problem? Here are two ways that a production manager might describe a problem:

   "We've missed our schedule three weeks out of the last two months because of Bob's sloppy work."

   "Bob's workmanship has been falling off. I know he can do better. I wonder what's bothering him?"

   The first approach suggests that making the schedule is the issue. The second uses the "people are our most important asset" concept.

Once you've identified these two sets of values, remember that both sets are important. The difference between the two is a matter of priority. It's part of a manager's job to allocate limited resources to best attain the company's goals. With unlimited funds, personnel, training, etc. you could achieve all of the goals. But you don't have unlimited resources. The mark of a good manager is devoting resources to the right priorities.

Social Norms and Rituals

Every society has unwritten rules of behavior. We greet people by shaking hands; the Japanese bow; the French kiss both cheeks; a soldier salutes. In the same way, the management culture has rules of behavior and rituals that must be followed if you are to be one of the group. Some of these are so common that they have become generalized rules, others are specific to a company.

Some examples of company-specific norms, from real companies:

In Company X, managers' offices have windows that look out onto the corridor. If a manager's door is closed, and you want to talk to him (or her), you stand by the window until you are noticed. The manager will either beckon you in or ignore you (in which case you come back later). To knock on a manager's closed door is a terrible breach of corporate manners, because it deprives the manager of his option to keep his private conversation uninterrupted.

In Company Y, it's a social gaffe to make lunch arrangements with someone before about 11 o'clock (except in cases such as birthday or retirement celebrations). The reason this rule developed is that the company has a fast-paced working environment, and most people don't know before 11 o'clock if they'll be available for lunch. Without conscious thought, employees developed this unwritten law to spare themselves the minor embarrassment experienced by both parties when a lunch engagement has to be canceled.

You probably follow most of your company's unwritten rules without ever thinking about it. Because they are so automatic, these rules may be hard to define. Again, the best method is to observe behavior, analyze it and filter out the irrelevant. The management subculture will follow some rules that apply to the whole organization, and others will be for their group alone.

Some of the more general rules:

1. Ever since John T. Molloy wrote "Dress for Success", most people have been aware of the importance of business attire. Many, however, have mistakenly taken the idea to mean that everyone should dress like an investment banker. Not necessarily. Each company has its own dress standards, both written and unwritten. There's usually one for management and one for everyone else. Watch what management wears. In some firms, cowboy boots with a suit are acceptable (especially in Texas), and in others, they look ridiculous. Other examples of this kind of thing abound. Look around you, and figure out what your company's dress code is.

   You have to watch yourself when visiting other companies. If you know their dress standards, dress to match them. If you don't know how your hosts dress, or if you're not comfortable with their style, the best bet is the consultant look: a conservative business suit.

   You may have heard of "personal image consultants". They will advise you to dress like the person who has the job you want, and some even recommend plastic surgery for "strategic career reasons". I can't deny that your appearance will have some effect on your progress through the organization. Studies have shown that, given equal qualifications on paper, people develop different perceptions based on appearance. Some common subconscious examples of this:

   People with eyeglasses are smarter and more serious than those without.

   Tall men are more forceful and effective than short men. On the other hand, tall women are sometimes seen as threatening by male executives.

   Blondes (male and female) are more friendly and outgoing than brunettes; redheads have volatile tempers.

Thin people are more efficient than heavy people.

To apply these notions to a hiring or promotion situation is a grave injustice, but most people don't even know that they're doing it. Prejudice based on personal appearance is a relatively unexplored subject; unattractive people have not been declared a disadvantaged minority group. The only case I can find where this issue made it to the courts is in the Christine Craft case.

2. Business entertaining can be as simple as lunch with a supplier or customer, or as elaborate as a Las Vegas stage show. A lot of deals are made (or lost) over a restaurant table. The rules for business entertaining are simple. First, suit the locale to the people. Some business contacts will be very formal people, while others will prefer a relaxed atmosphere. If the guest is from out of town, ask if there's a regional specialty they want to try.

   Second, when in doubt, err on the side of good taste. Topless bars, punk music clubs, and anyplace where the food comes in little Styrofoam containers with cartoon characters on them are generally not a good idea.

   The third rule of entertaining concerns alcohol. Know your capacity, pace yourself, and quit while you're ahead. The movie stereotype of the businessman who gets plastered on his expense account is not the model to follow.

   Another question is "who picks up the check?". If one party issued an invitation, they are presumed to be the host. If it's unclear whose idea the get-together was, there's an implied rule that the party who's trying to sell something will get the check. If this doesn't apply, you should at least offer. As a corollary, never go to a business lunch without at least one major credit card.

   The final rule is No Free Lunch. People who schedule lunch meetings so they can stick the company with the check will be found out someday. Is a free lunch worth your job?

3. If you attend trade shows conferences, you've seen vendors giving away small items with their company logo on them. A certain amount of this is fine; it's legitimate advertising. Remember, though, that if gifts become lavish or expensive, they begin to look like bribes. Most companies have a set policy on the acceptance of business gifts. Read yours. Also, remember: cash is never an acceptable business gift.

   Gift-giving is uncommon in business, aside from promotional items. But in Japanese society, the giving and accepting of gifts between business associates is the norm. When dealing with managers from Japan, be prepared with a gift suitable to that person's rank in his company. Regional items are popular with the Japanese; on a visit to Texas, a Japanese client was delighted by Stetson hats and western boots.

4. Sometimes it seems we spend most of our day in meetings. That being so, a few guidelines are in order. First, minimize meetings. Only call a meeting when you can't deal one-on-one with those involved. Issue written notices to all those invited, with the date, time, place and an agenda. At the meeting, follow the agenda and give everyone a chance to speak. The person who called the meeting is responsible for controlling the flow of discussion. Keep the talk to the point, it'll make your

meetings shorter and more effective. Lastly, after the meeting send the attendees a summary of what was discussed, what was decided, and the "action items" -- who promised to do what. This will prevent misunderstandings and provide a record of decisions.

5. The last item in this section is not peculiar to management, but it is critical to the success of a manager. That rule is to exercise tact and good manners at all times. It costs you nothing (except a little emotional strain) to be calm and polite when faced with fools, knaves, boors and other unappealing personalities. Never "bad mouth" company personnel, customers or the competition. That's right, the competition. It's OK to compare products on a factual basis, but don't describe them as crooks or incompetents.

The socialization process is difficult. You have to adjust to the ways of a new group without losing your individuality. Your peers may object to your conduct; they may see it as pandering to the boss. The hardest part is that, if you're trying to get promoted into management, you have to begin the socialization process without being a member of the group. Your goal is for managers to see you as part of the group because you behave like a manager. When you begin this process, you're still an outsider to the management subculture, and, like all subcultures, they will resent the "invasion". Go slowly. If you sense discomfort in a situation, back off. You don't have to do it all before you get promoted; new managers are expected to be a little "green".

STEP THREE: FAMILIARIZATION

The third step in the process is to make management familiar with you, in the sense that they know who you are, they know what you can do, and they know that you want to move up. Don't take this for granted. People in technical positions often go unnoticed by managers outside their own departments. We all tend to classify people; at work, job titles make it even easier. If you're pigeonholed as a programmer, and you want to be a manager, you have an "image problem" to overcome. You need a "publicity campaign" that will raise your profile without making you look like a junior-grade Machiavelli.

It's not enough to say "I want to be a manager". There are a lot of management spots out there; some you'd love, and some you'd hate. You need to define your ideal position:

Large, medium or small size company?

What kind of business? Manufacturing, retail, distribution, services, etc?

What function do you want to manage? Personnel, production, field service, etc? You don't need to target the DP slot. If you truly understand your company's accounting system (for example) you might make a good accounting manager.

Where in the organization? Would you rather be a branch manager, or one of several department heads (on the same level) in the main office?

What kind of path do you want to follow? If you want to be a purchasing manager, for instance, it might help to work in or with Accounts Payable, Receiving, and Warehousing. Very seldom can you head straight for the top.

In every company, there are people who exercise power, and not all of them are in upper management. Power, in this sense, is the ability to make decisions that affect the business, and to influence the decisions of others. Someone once said that anyone can say "no", but only the powerful can say "yes". Look for the people who can get things done on their own authority.

The idea of identifying centers of power and managing with those centers in mind is distasteful to some people. It seems like "currying favor" (there are less polite expressions). If you manage with the sole purpose of impressing these people in order to advance your own career, then you are indeed currying favor. But if you act with the realization that these people influence the company's fortunes, and have a right to the best input you can give them, then you are performing your duty as an employee.

So how do you "manage upward" in a positive, productive and non-self-serving way? Here are a few suggestions:

1. If you're providing input to people not in your own chain of command, keep your boss informed. He (or she) shouldn't have to wonder if you're working for other managers.

2. Give credit where it is due. Just because you presented an idea to management doesn't mean it's your idea.

3. Keep the communication two-way. Ask the key people (not just your own boss) about their priorities and problems, and make these your key items.

4. When you're wrong, admit it and take your lumps. It takes a person of character to say "I blew it, but I learned something, and I won't make the same mistake twice".

5. Avoid office gossip. Sooner or later someone's going to repeat what you said about some other guy, and it'll all come back to you in a most unpleasant manner.

6. There are times when you should rock the boat. A Yes-man attitude may pay short term dividends, but in the long term, it may be better to have a few unconventional ideas. Just make sure that when you put forward a new suggestion you can back it up with business reasoning. Like any organization, a company's management can get set in its ways. A little fresh air is needed from time to time.

7. Find a mentor. Some companies have formal mentor programs to develop managers. If yours does not, identify someone a couple of levels above you with a compatible personality. If you make a point of asking their advice, informally, on business matters, you can develop a relationship that can serve you both. You will gain knowledge and a friend in management, and your mentor will enjoy helping you along.

Never make yourself unpromotable. Many of us know a DP person who is the local "guru" for a specific application or system. Some people think that this is "job security". It may be, but only if you want to do that job for the rest of your career. As you plan for your own advancement, "groom" someone to take your place. When your chance comes, you'll be available to take it.

Let your supervisor know that you want to move into management. Some people dislike this idea; they're afraid the boss will think they're "gunning for his job". Not necessarily. For one thing, the idea applies to your boss, too. If he wants to be promoted, you might find yourself being "groomed" for your boss' job. Besides, there are more paths than one to advance up the organization. Your boss may know of an open spot for you.

Sometimes the way to move up is to move out. If there is no path upward in your company, you may have to look to the employment market to realize your career objectives. This is something your boss may not like, but you owe it to yourself to pursue the path that will make the most of your abilities. This doesn't mean that you should "jump ship" every time the opportunity arises to make a few more bucks. It simply means that you should not rule out the possibility of advancing by changing companies.

WHEN YOU GET THERE

Ability, qualifications and length of service are not a guarantee of advancement. Never get the attitude that someone "owes" you a promotion. All you can do is prepare yourself, watch for an opportunity and seize it when it comes. To act as if a promotion is something to which you are entitled is an easy way to antagonize your superiors into not giving it to you.

I regret that "The Do-It-Yourself Career Advancement Kit" carries no guarantee, either. The ideas and techniques discussed here can improve your chances, but that's all that I can do for you. You really do have to "do it yourself".

A Manager's job is not giving orders. The best managers never find themselves in a position where they have to say "do it or else". For these managers, it is enough to assign the task; they know it will be done. The only way to achieve this state of cooperation is to live by the following rules:

1. Keep your people informed. Let them know how the company's doing, what's coming up for the department, and other matters of general interest. By showing your people that you want them well-informed, you help them identify with the company as a whole, you increase their self-esteem, and you get a valuable opportunity for feedback.

2. Listen to people. Communication flows in all directions. If you add up all your subordinates' experience, it probably totals several times your own. Encourage them to make suggestions, and to question what they don't understand. The final word is yours, but your people should know that their input will be given serious consideration.

3. Be honest with your people. If there is something that (for policy reasons, etc.) you can't tell them, say so. If you ever tell them a lie, and they find out, they may never forgive you, and may never trust you again.

4. Be approachable and agreeable, but never get emotionally involved with subordinates. One of the biggest mistakes new managers make is trying to be a "buddy" to subordinates (the other big mistake is coming on like Genghis Khan). There's a fine line to be walked, and the only way to find it is by experience. While you're learning it, apply an accountant's rule and always err on the conservative side.

5. Be absolutely fair. As a manager, you'll have control over work assignments, schedules, opportunities for training and travel, and other variables that affect the quality of work life. Your objective should be to utilize each person's abilities, while giving everyone an equal chance at the enjoyable aspects of the job. This is another area where mistakes are seldom forgiven. The programmer who's done nothing but maintenance for a year will be resentful, and you may never be able to make it up to her (or him).

6. The best thing you can do to get work out of people is to make the work as interesting as possible. Given a problem to solve, the proper tools, no administrative obstructions, and the knowledge that their efforts are important to you and to the company, most people will grab that problem by the throat and get down to business.

CONCLUSION

If you want to get into management, you can't sit around and wait to be tapped on the shoulder. You must develop the skills, learn to interact as a manager, and make your goals known to those who can help you advance. There's no guarantee of success, but you will never get there if you don't put some effort into it. If you want it done right --DO IT YOURSELF.

Anarchy in Management

A Survivor's Course

Louis R. Mills, CDP

Bio-Rad Laboratories, Inc.

1000 Alfred Nobel Drive

Hercules, CA 94547

(415) 724-5613

How much anarchy is there in your organization? Anarchy is not the same as chaos. Anarchy defines a management style and is not necessarily a mass of confusion. Let's begin with what we're going to cover:

1. The basics of organization theory.
2. Two contrasting models of organization.
3. Decision systems within these models.
4. Influencing decisions within these systems.
5. Some conclusions.

The basics of organization theory rest on:

1. The organization's goals
2. Its technology base
3. Its hierarchy or structure
4. Its environment
5. The setting (strategic or sub-unit)

The two contrasting models of organizational behavior and structure are:

1. The Classical Bureaucracy (Weber)
2. The Organized Anarchy (March)

What is a Classical Bureaucracy? It can best be understood using the following terms:

1. Its goals are clear and well understood, inside and outside the organization.

2. It has an unambiguous technology base.

3. Pyramid reporting structure (most likely).

4. Participation is stable.

Good examples of such bureaucracies are: armies, churches, industrial giants, hospitals.

What is an Organized Anarchy? Using the same criteria:

1. The goals are not clear or well understood.

2. The technology is ambiguous or broad spectrum.

3. Flexible reporting structure (or vague one).

4. Participation is fluid.

Good examples are: colleges and universities, high-tech companies, agencies containing elected officials.

How do decisions get made in these organizations?

1. The Rational System (Fayal, Taylor)

2. The Garbage Can (March)

Rational decision making includes:

1. Defining the problem (product < goal)

2. Establish criteria for solution adequacy

3. Generate alternate solutions

4. Evaluate solutions against criteria and make the optimum choice.

5. Evaluate the effectiveness of the solution (product > or = goal).

We use rational decision making all the time in data processing. For example:

1. Problem. Terminal response time too slow.

2. Inquiries must respond within 10 seconds.

3. Upgrade processor, spin off applications, etc.

4. Select new data communication equipment.

5. Response time improved from 30 second average to 15 seconds.

Other examples of rational decision making include methods like Zero Based Budgeting, etc.

The Garbage Can process is interesting:

1. A decision opportunity exists

2. Appropriate resources exist

3. A decision is made

The best example of this is the "water cooler" model of decision making, but also includes hallway chats, staff meetings, disciplinary conferences and goal setting exercises.

How do you influence decisions within these contrasting models?

Under the Rational model, you must:

1. Be in the appropriate position within the structure.

2. Be able to influence an individual who is in the appropriate position within the structure.

In other words, nothing gets done outside the formal channels. You must either have the authority or convince the authority to decide. If you don't have the authority yourself, you must become adept at influencing those at the top. Examples include: presentations to the Board of Directors, the Executive Dining Room, the Country Club membership.

The Garbage Can model is very different. There are several ways to effect or influence decisions:

1. Spend time

2. Persist

3. Exchange status for substance

4. Facilitate opposition participation

5. Overload the system

6. Create decision opportunities

7. Manage unobtrusively

8. Interpret history your way

Captain Grace Hopper, the mother of COBOL, tells us, "it is far easier to apologize than it is to get permission." Remember, if you were only trying to help, others will feel bad about censoring your mistakes. But they won't forget them.

Some conclusions to take back to the office:

1. Organizations vary between the two extremes.

2. Your ability to influence may be significantly independent of the organizational setting. You can get things done.

3. Manage your risk. Make decisions that have good payback to you and your staff.

4. Know who the other decision makers and influencers are. Use them.

EASING THE STRESS DILEMMA

by
Phyllis S. Kramer, M.S., M.F.C.C.
Engineered Software
856 North Monterey Street
Alhambra, California 91801
(818) 570-8028

Definitions of Stress

Webster's dictionary defines "stress" as strain; mental or physical tension; pressure

Some other definitions are:

Any action or situation that places special physical or psychological demands on a person

Anything that can unbalance the individual's equilibrium

Depleted energy with lower resistance to illness plus increased dissatisfaction and inefficiencies

Response of the body to any demand

Discrepancy between what you want and what you are getting

Causes of Stress

In our culture stress is a factor in everyone's life. The power and importance of it can no longer be ignored.

Merely being human predisposes us to generate stress. Humans are equipped with the ability to look ahead, the capacity to control some events in the environment, and the imagination to think of "what might have been". All these things can create stress.

And then there is also environmental stress that can affect us all. Some of the major causes of environmental stress are: crowding of people (which tends to create a loss of control), noise and air pollution. It is interesting to note that "traffic" can include crowding, noise and air pollution all in a single situation.

Whenever there is a crisis situation (unexpected dangers), there will be stress. The experts say that, whenver possible, it is best to "warn" an individual. The "warning" serves as a way for the person to relieve the mental stress a little at a time prior to the actual event.

Isolation from other people has been known to cause such stress on individuals that it can create long-lasting detrimental effects.

Akin to environmental causes are the possible sociological ones. Among the strongest of those in our current civilization are loss of tradition and changing family structures.

The famous "Social Readjustment Rating Scale" developed by Holmes and Rahe (and listed below) lists a large variety of stressor agents.

Rank	Life Event	LCU Value
1.	Death of spouse	100
2.	Divorce	73
3.	Marital separation	65
4.	Jail term	63
5.	Death of close family member	63
6.	personal injury or illness	53
7.	Marriage	50
8.	Fired from job	47
9.	Marital reconciliation	45
10.	Retirement	45
11.	Change in health of family member	44
12.	Pregnancy	40
13.	Sex difficulties	39
14.	Gain of new family member	39
15.	Business readjustment	39
16.	Change in financial state	38
17.	Death of close friend	37
18.	Change to different line of work	36
19.	Change in number of arguments with spouse	35
20.	Large mortgage	31
21.	Foreclosure of mortgage or loan	30
22.	Change in responsibility at work	29
23.	Son or daughter leaving home	29
24.	Trouble with in-laws	29
25.	Outstanding personal achievement	28
26.	Wife begins or stops work	26
27.	Begin or end school	26
28.	Change in living conditions	25
29.	Revision of personal habits	24
30.	Trouble with boss	23
31.	Change in work hours or conditions	20
32.	Change in residence	20
33.	Change in schools	20
34.	Change in recreation	19

35.	Change in church activities	19
36.	Change in social activities	18
37.	Moderate mortgage	17
38.	Change in sleeping habits	16
39.	Change in number of family get-togethers	15
40.	Change in eating habits	15
41.	Vacation	13
42.	Christmas	12
43.	Minor violations of the law	11

This scale has been used to successfully predict the tendencies toward illness and dysfunction that may be expected in an individual as a result of the various stressors.

Currently there is a new type of stress to be added to all the others and that is the stress that has come as part and parcel of the computer revolution. In his book Technostress: The Human Cost of the Computer Revolution, Craig Bond describes some of those factors as the removal of human contact, the expectation of people that they be as fast and accurate as the computer plus the internalizing of the machine model and attempting to become more like a computer (which has the end result of people being less creative).

For those working directly in the field of computers, an added stress is the need to stay on top of constantly evolving technology.

While the causes of stress that are described above are universally applicable, many causes are specific to an individual.

One of the largest factors in an individual's stess can be that they have the wrong job. If the type of work that a person is involved in is not rewarding, stress will permeate

their entire life. It is indeed a fact of life that no amount of recreational activities, no amount of outside interests, no amount of money can make up for the personal deprivation of work without pleasure. Connected to this deprivation is usually a strong feeling of self betrayal and conflict of values.

While part of the reason that people remain in the wrong job may be fear of trying something new, another part (particularly in the computer field) is explained in the term "golden-handcuffs". Higher-paying jobs tend to hold people there even when the work brings no pleasure into the person's life.

Another cause of individual stress can be that, even though the work that the person has chosen is appropriate for them, they are working for a company that isn't. A (very) few of the many situations that might make a work environment the wrong one for an individual are

- disagreement with company policies
- unchallenging assignments
- too many bosses
- work overload
- not enough work
- job insecurity
- time pressures
- poor work relationships

Specific to poor work relationships would be a poor relationship with one's boss. There are some types of bosses that are going to create a poor work envirnoment for any and all people that have the misfortune of having to report to them. Easily identifiable ones are

The "buddy"; a boss that is extremely friendly but doesn't define jobs/tasks. This boss wants love as well as the avoidance of pressure
The boss with unrealistic work expectations
The critic
The boss who has "gone fishing" (i.e., who has retired on the job)
The uncommunicative boss
The survellience master
The boss who is incompetent in his own job

Sometimes even the best choice of work can become extremely stressful if it has become what could be classified as a "dead-end" job. A few characteristics of a "dead-end" job might include

- no recognition
- no challenge left
- freedom restricted by company rules
- no hope of increased earnings and/or other rewards
- advancement prevented by promotion barriers

There are some jobs and/or life-styles that are so full of schedules, clocks, watches and deadlines that the time pressures are enormous.

Those that suffer from an obsession with "perfection" are setting themselves up for un-ending and debilitating stress. While concern with quality is definitely a legitimate and commendable one, the need for perfection concentrates so much on small details that the overall task suffers from it. Since absolute perfection is almost always an impossible goal, the constant striving (usually with much difficulty) for something that cannot be attained is an unhealthy situation.

There are also a wide variety of attitudes involved with interpersonal relationships that cause stress. Some of these are

- other peoples unwillingness or inability to understand or change
- attempts to please too many people
- frustration with the fact that people will disappoint you

A great deal of the frustrations with other people comes from the difficulty most people have in integrating the fact

that different environments and methods of raising children create widely differing values and outlooks on life. When an individual expects others to interpret and/or react to things the way they do, they are bound to add a fair amount of frustration into their lives.

Another area that can generate stress is that connected with financial concerns. At the level of the individual, this would mean not those financial matters that would be stressful for anyone, but rather those that are specific to the person. An example of this might be that for some individuals having no "rainy day" savings would be of no concern while for others it would be a situation inducing constant worry. Financial concerns are another area where the type and amount of stress can (and does) vary a great deal from individual to individual.

While no doubt the reader can think of many other causes (or potential causes) of stress, the last that we will take the time to mention in this section is one that has an influence on us all at some time in our lives. That stress is the one arising from the difficulty in accepting the fact that life is NOT always fair or just.

The Results and Costs of Stress

Both the psychological and physical results of stress run from the slight to the extreme in intensity and are exhibited in an almost infinite variety of symptoms.

A few of the more general and milder personal results of stress are muddled thinking, anxiety, and fatigue.

The stress of the computer revolution has created another common result/cost in that people are becoming less able to communicate and interact with other human beings.

As stress becomes more and more severe, the amount of illness increases dramtically. The medical community now estimates that somewhere between 75% and 95% of all disease and illness are either a direct result of stress or an indirect result of it.

In addition to the increase in physical problems as stress becomes more severe, the psychological problems such as impaired relationships (including divorce) and lowered self-esteem become major issues.

The work related costs of stress figure in the billions of dollars yearly and include such things as lower productivity,illness, absentee-ism and premature employee deaths. The costs of the premature employee deaths alone are estimated to be as much as the combined profits of Fortune's top five corporations.

A recent five-year study of executives from fourteen major corporations directed by Dr. Kenneth Pelletier of the University of California at San Francisco Medical School revealed that 93% of those executives suffer from life-threatening stress. This 93% not only suffered from ill health, but also from impaired relationships. Very few were able to enjoy their professional success.

And then there is also the ever present danger of job "burnout". One of the basic facts about burnout that some people have difficulty in accepting is that NO one is immune. It can and does happen to individuals in all levels of the work force.

Some of the classic signs of job burnout are:

1. increased use of food, alcohol, cigarettes or drugs
2. exhaustion
3. lack of interest, not caring, detachment
4. negativity, hostility, paranoia
5. physical ailments

One important point to keep in mind in connection with work burnout is that even though too many hours of work can cause job burnout, the stress related to the job can be even more significant than the number of hours.

A new phenomemon has developed in the computer age with the development of a brand new personality type referred to as the "technocentered" personality (Technostress: The Human Cost of the Computer Revolution by Craig Bond). This person resembles the obsessive-compulsive in that they have trouble managing time and/or working within schedules. They lose themselves in detail. Along with their intense power of concentration a rather strong fear of losing control is also evident. The difference between the technocentered person and the obsessive-compulsive is that where the later is preoccupied with self-doubt, the former is not. The tecnocentered person has no sense that something is psychologically wrong and they interpret their inner pressure as a desire to excel. They cannot articulate feelings and are usually difficult to deal with. They see themselves as flexible, but their flexibility is really an effort to avoid conflict and to maintain as little human contact as necessary.

For the technocentered personality to make a breakthrough in understanding either themselves or others it is (unfortunately) usually necessary for a personal crisis to occur in their lives.

The special demands of computer work exacerbate obsessive-compulsive tendencies without allowing for humanizing influences, such as ample social contact. Accelerated performance becomes the overriding goal and the ability to arrive at creative solutions is lost.

The list of results and costs of stress could be even more numerous than the number of individuals on the planet, but in closing this part of the discussion we will look at the results of stress in its most extreme form.

In some cultures stress has been used to actually cause death. Extreme stress, deliberately applied, has been used for ritual execution in places as far apart as Australia, Borneo and the islands of the Caribbean Sea. It is the extreme punishment for the breaking of a sacred taboo. A witch doctor often merely points a "magic" stick at the transgressor and/or recites fateful incantations that place the victim under a spell of death. And death does come, usually in a matter of days. Harvard physiologist Walter Cannon made a study of these deaths by suggestion and reported extensively on some 30 "voodoo" deaths. While they all consisted of a number of different elements and approaches, the common denominator in all of them was the withdrawal of tribal support. This is an interesting fact to keep in mind when we think of the alienation that so many people feel in their lives today.

"Alleviating Stress on the Job"
an article by Phyllis S. Kramer

Reprinted from Vol. XXIII -- No. 42 of
COMPUTERWORLD

One of the keys to dealing with stress is to be familiar not only with the common causes of stress, but also with those stressor agents that are specific to you as an individual. Once the causes have been determined, then it begins to be possible to develop techniques and strategies to prevent and/or manage the stressor agents in your life.

Another key is to become familiar with your own signs of stress when they are still slight and deal with them then rather than allow them to build up to a level that is more difficult to manage.

Specifics of alleviating stress might begin with analyzing whether or not you have the "right" job. If you are doing work that does not suit you, the amount of stress will eventually build to a dangerous level. People who truly enjoy their work have a great deal of energy and enthusiasm for life, so if you find that you are feeling tired and/or unenthusiastic it might be appropriate to consider whether or not the "wrong" job might be the problem.

Beware of the "golden handcuffs" syndrome where you get yourself seemingly "locked" into a job that has financial benefits and/or security but that gives you no pleasure. So much of our lives is spent in connection with our jobs that how we feel about the work we do can affect every aspect of our lives.

It is valuable to be reminded that there is often a "catch-22" about work and the amount of income that goes something like this: usually people who enjoy their work do their job better and earn more money. The catch is that they do their job better partly because they are not primarily interested in the money.

Changing careers is often a difficult thing to do, but the personal cost of continuing to do work that you don't enjoy is more than any individual should ask of themselves. Consider the possiblitly of trying out potential new careers as hobbies and/or in a moonlighting mode. Sometimes a modification of current work is all that is needed for the work to move from being un-rewarding to enjoyable. If you do consider a career change, there are a large number of books available on the subject and career counselors to assist in the process.

Even if you are doing work that you enjoy, it may be that you are not in the "right" company. While some companies may not be a good place for anyone to work, it is also true that the type of company that is appropriate for any individual is indeed an individual matter. In order to determine what type of company you want to work for, some thought on the subject will be required. If your current situation does not meet your requirements, then it may be time to move on. However, if you do move on, you should first be clear on what type of company you are seeking and then do your best to determine if the companies you are considering do or do not match your own needs. In evaluating companies, it may be necessary to prioritize your requirements in that possibly no company will be able to meet ALL your needs. It is very important, however, not to compromise on the issues that are important to you or you will find that you have just moved to another "wrong" company.

In discussing the "right" company, it is again necessary to consider the trap of the "golden handcuffs" and not be caught up in it.

And if there seems to be no "right" company for you, perhaps it is time to consider going to work for yourself.

Once you have the "right" job and the "right" company, a further way to alleviate stress would be learning and using some of the job-related skills that are notoriously good at preventing or alleviating stress.

An excellent aid not only to alleviating stress, but also to maintaining one's general health is exercise. Exercise does not need to be gruelling or an obsession for it to be very beneficial. In fact, if the type of exercise chosen is distasteful to you, you might need more of it to get the same benefits as you would from less of a more enjoyable type of exercise. So try to make whatever you choose for exercise some type of activity that you enjoy. Also remember that even a small amount of exercise can assist in reducing stress ... thus a five minute walk is well worth the effort.

Along with exercise is the value gotten from deep breathing. A few minutes of this during or just prior to a particularly stressful event can change the entire complexion of the event.

The concious and methodical relaxation of muscles throughout the entire body has long been known to be good for stress reduction. Biofeedback is a extensive use of this technique, but the technique can be used in a much more casual atmosphere and without the use of special equipment. There are many books on the subject as well as audio tapes that will lead you through the steps found most useful. Once the technique has been learned, the books and tapes even become unnecessary.

A healthy diet is essential for alleviating stress as well as for preventing the addition of more stress in one's life. Most people are aware of the basic requirements for good eating habits (including the removal of sugar, fat, salt and caffiene from their diet), but it is usually necessary to keep reminding ourselves to practice what we know.

One of the healthiest things for both our minds and our bodies is laughter. Numerous medical and scientific studies have proven that endorphins are actually produced by the brain during laughter. That means that not only is it a great way to relieve tension, it is a bona fide pain reliever as well. Try to think of ways to add this to your daily life.

In light of the fact that traffic is a very strong stressor agent, it is a valuable idea to have things available to listen to whenever you are going to be in traffic. A variety of relaxing music, "waves", tapes (or a radio program) that generate laughter or interesting stories are a few of the possiblities.

It is always a good idea to periodically clarify your values and your goals in life. Once that is done, then it becomes possible to more easily establish priorities and to do the sometimes difficult task of eliminating some activities from your current workload and/or life.

When the current load seems overwhelming, then it is best to set up short term goals and to focus only on those for awhile. Also helpful is taking the time to write down your own personal "stress inventory". Just focusing on the issue enough to put things down on paper can be enough to jog the mind into seeing ways in which some of the stress in your life might be eliminated.

Once some thought is given to the topic, it becomes apparent that there are many ways to prevent, control and alleviate stress. The "trick" is to begin the process by MAKING the time to give the issue some consideration and to relate it to your own life.

It is also essential to remember that some problems are NOT solvable. You can only control YOUR life and hoping for/waiting for others to change is not realistic. To be successful, any type of changes in your own life must be self-motivated.

Even when under your own control, change is a slow process and usually requires a great deal of practice. When instituting change, it is best to take small steps. Keep in mind that sometimes when a change does occur, it may not be immediately obvious. Remember,too, that any amount of change in the right direction means that you CAN, with persistence, reach your goals and make your life the happy, healthy life that you deserve.

Bibliography

"Anatomy of an Illness"
by Norman Cousins

"Health is a Question of Balance"
by P. Brenner

"Mind as Healer: Mind as Slayer"
by K. Pelletier

"Minding the Body, Mending the Mind"
by Joan Borysenko

"Modern Madness: The Emotional Fallout of Success"
by D. LaBier

"Music Therapy and Stress Reduction Research" by S. Hanser in "Journal of Music Therapy"; Vol. 22(4), pp 193-206; 1985

"Necessary Losses"
by J. Viorst

"Occupational Stress: A Significant Contribution to Effective Utilization of Technology" by Salvendy in "International Journal of Productivity Research"; No.3, pp 409-416; 1982

"Professional Job Changing System"
by R. Jameson

"Psychological Immortality"
by J. Gilles

"Reconciliation: Inner Peace in an Age of Anxiety"
by T. Rubin

"Stress"
by M. Whittlesey

"Stress Disease: The Growing Plague"
by P. Blythe

"Stress Seen Hobbling Productivity" by P. Keefe in
"Computerworld"; September 1988

"Technostress: The Human Cost of the Computer Revolution"
by Craig Bond

"The Dark Side of Megatrends" by K. McClellan and
M. McClellan in "Employee Assistance Quarterly;
Vol. 2(2), pp 1-29; 1986

"The Depression-Stress Link" by Christopher Vaughan in
"Science News"; Vol. 134, No. 10, pp 145-160;
September 3, 1988

"The Stress of Life"
by H. Selye

"The Work/Stress Connection: How to Cope with Job Burnout"
by R. Veninga and J. Spradley

"Women Still Battle Barriers in MIS" by J. Savage in
"Computerworld"; September 1988

"What Color is Your Parachute"
by J. Bolles

Selling to Decision-Maker Buy-Off
or
Managing Up the Hierarchy

Mitchell Kleiman
Ernst & Young
One Sansome Street
San Francisco, CA 94104
(415) 951-3205

Abstract

Do the first 5 words out of your mouth put your executive to sleep??

another way of saying this is,

Do you keep explaining the details involved in a purchase decision, even as you watch your manager's eyes roll upwards into his or her head?

Do you waste short audiences with your manager?

or

Do you use your rare (15 minute) meetings with your executive to explain detailed plans and to "educate" him or her (whether or not they want to get educated) on how some new technological gadget will help him or her?

The questions above point to a style of speaking, writing, and presentation that runs rampant in technical fields. So much detail may not be necessary. In fact, it may be too much.

Making presentations to management is a crucial aspect of the technical managers' job. These presentations impact key issues such as the budgeting of resources (capital and personnel) and strategies affecting the future of the technical function.

Presentation and listening skills are a key determinant in how far and fast a technical manager's career may rise. Yet most managers lack basic training in giving presentations, feel uncomfortable, and avoid presentation situations.

This paper is aimed at expanding your comfort zone and understanding, as well as enhancing your ability to interact with decision-makers in your organization.

Introduction

Wouldn't it be wonderful if you got along with your executive and knew exactly where you stood with them?

Technical managers who have worked their way up through the programming ranks often find it difficult to transition to a managerial style. While technical ability and attention to detail are critical qualities in analytical positions, as a manager, the critical skills are the ability to prioritize projects, summarize information, and communicate. The technical manager is called upon not only to manage the technical resources, but to represent that functional area to the executives of the company and steer the group through potential political pitfalls. These are new arenas for the technical manager to be in - often they are interacting with people they rarely if ever saw before and are unfamiliar with.

Communication is one of the most important, if not the most important skill necessary for a competent manager. It plays a critical role in the relationship between the manager and his/her employees. It plays an even more crucial role for the manager in his relationship with his manager (or executive). This is because a manager typically interacts up the hierarchy less than they do down.

Communications between a technical manager and an executive in written form often are limited to memos, reports, and project proposals. Verbal communications, besides phone calls, typically are face to face meetings and presentations, and management group meetings. These communications impact the budget of the technical service department, number of personnel, projects, and it's reputation.

Communication and "managing up the hierarchy" are areas that many people are uneasy with and that many people are not educated in.

Sales professionals are typically trained in communication skills, strategizing, and understanding people so as to be more effective in selling to people. The use of sales training and techniques in interacting with executives will make you more successful and can increase your success in dealing with sales and marketing professionals in your organization.

Basics

There are three basic points that will be discussed in this presentation, that are intended to improve your ability to and your comfort level with, communicating with your executive and "selling" them on your ideas.

The basic points are, as follows:

1. Executives are people too
2. Design your communications to work for your executive
3. Get to the point (or Lose the details).

While these may seem simple and obvious, as we go through the presentation, you may want to examine if your interactions with your executive are affected by some of the things we discuss.

Executives Are People Too

Some technical managers are uncomfortable around their executives for a number of reasons: age, different backgrounds and education, community status, and length of service with the organization. A manager may feel pressured to respect the executive. Often the manager is interested in establishing a working relationship in which he/she understands the executive and works well with them. In addition, the manager's personal performance evaluation and the success of their department is to some extent in the hands of the executive.

Executives Put On Their Pants Just Like Everyone Else

I fine that staff people as well as managers sometimes are in awe of executives, and may be intimidated. When I did some research into the nature of executives, I found that they appear to be human - they "put on their pants" just like everyone else, ... and they happen to have be within the organization. They are not necessarily as a group smarter, friendlier, more astute, or more hard working than your average company person. They are however your manager. They are human - they sometimes make mistakes, sometimes change their minds, and they may do things they regret later. Just like you and I.

Build A Relationship

In working with various executives over the past twelve years, I have found it beneficial to work on building a relationship, rather than just working together. This results in mutual respect and trust, and serves as a foundation for resolving difficulties in the future. As executives spend most of their time interacting with people about projects and tasks, many tend to person-oriented as opposed to task oriented. They want to know that they can rely on you, what motivates you, and that you are capable of handling the job. The building of this relationship often requires the manager to prove himself to the executive, delivering quality work, on schedule. As the trust develops, the manager is often given more leeway to make his own decisions, increased responsibilities, and more opportunities. It is critical in the early stages of building a relationship to say what you are going to do and then do it. So you may want to think carefully about you say.

Make it Easy For Him To Like You

Building the relationship is one of the things to do as part of interacting with your executive as a person. Besides having your materials and information ready for a presentation or meeting, also be prepared to discuss topics of interest to the the executive, have a story or joke ready, or ask them how their weekend or executive retreat was. Make it easy for them to like you. Read up on the topics of interest to the executive, or if you note an article in the newspaper or in a technical journal that might be of interest, forward it to them with a note. This demonstrates your commitment to working with people and sensitivity to the interests of others.

Make Him A Part Of Your Team
This contributes to having the executive as part of your "team". When you present ideas or projects to the organization it will be valuable to have the respect and support of your executive. If sides need to be taken, you want as many people as possible on your side, and the higher up in the organization, the better.

Climate That Will Give Your Idea The Best Reception
There are often times that are better than others to talk with an executive about a project or idea. For example, financial executives are usually very busy and preoccupied at monthly, quarterly, and yearly closings. Keeping your meetings brief at those times and scheduling around those busy times may get your ideas a better reception. Check for a climate that will give your ideas the best reception.

Path of Least Resistance
As is true for many people, executives often prefer to pursue the path of least resistance. Whenever possible, design your projects and ideas to be as close as possible to the path of least resistance to increase their chance for acceptance and success.

Design Your Communications To Work For Your Executive

In the normal course of events, a manager communicates with executives in meetings and through memos. The memos as a physical form of communication, are in existence longer and may also be more widely distributed. You cannot always be sure who will see a memo or written project report. Therefore, extra care should be taken with written communications to insure they get the message you intended across.

Besides concerning yourself with developing a clear writing style and focusing on issues, it is useful to consider your audience and design your communication for them. If the audience is only concerned with one aspect of a project, focus on that aspect and minimize the "noise" or other information you provide them.

Communicate, Don't Manipulate

When putting together proposals for management, I would focus on building an air-tight case for my proposal. When I would present this information to my executive and lead him through my analysis I noticed that sometimes the person would get uncomfortable. One of the main job functions of an executive is to gather and analyze information and make decisions. As a manager, you should suggest courses of action based on your analysis but you may want to avoid forcing decisions upon the executive. Many people dislike decisions forced upon them. Develop your idea, present your analysis and recommendation. Be prepared to present other alternatives, their pros and cons, and why you think this is the best one. Then leave it up to the executive to make the decision. Empower the executive with information.

Put The Reins In Their Hands

Some executives are unfamiliar with technology and may be concerned about the cost of technology within the firm. As has been often reported, technology projects often run over budget and from time to time cause havoc in a company. Executives have legitimate concerns about how to manage the technology function.

It is crucial to develop methods of communicating basic data processing functions so as to provide the executive responsible with a way of understanding and dealing with changes in the business environment. For example, developing revenue per technical person or technical services rendered (in dollars) per technical person figures will give an executive a way to gauge what they are receiving and how to adjust to increases or decreases in business, service levels, and changing internal conditions. The development of charts that indicate hardware and software usage, costs, and levels of service that are understood by the executive build an understanding of what the technology function provides and what it accomplishes. In addition, the implementation of productivity measurements allow the executive to gauge what you as a manager are accomplishing. Together, these tools provide you with a way to communicate what you do, what it costs, and how well it is going.

Prepare Your Proposal For Success - Structure It For Their Comfort & Understanding

As you work with your executive, learn from their comments and feedback how they like to have memos and reports presented to them. Do they prefer written reports or face to face briefings? Do they want a summary report or to have all the detail available at a moments notice? Do they always want a financial analysis prepared prior to meeting with them, or after a decision has been made? Are they comfortable
making a decision based on your recommendation, or do they like to have a second or expert opinion on the matter?

As you notice these items, incorporate them into your planning, meetings, and presentations so that the executive is more receptive to your ideas and projects, and you have an increased chance to succeed. Learn to do it the way they like it.

Learn Their Language

Another way to increase the comfort level of the executive to your ideas and projects is to structure your communications in their "language". As people develop expertise in particular areas, they learn new terms that are more precise in that environment and they use common terms in different ways. An outsider of that environment may not understand the jargon or may find it offensive or boring.

Minimize your speaking "computerese". Translate the technical jargon into English so that you do not bore them.

Many executives come from the financial, strategy, or marketing side of the company. Learn to communicate concepts in terms they use and understand. If you are not versed in the field of their expertise, if possible take a course in accounting (e.g. local colleges usually offer accounting for managers or for non-finance people). This knowledge will expand your skill base and provide some common ground with your manager.

Be Prepared

Before meeting with an executive (or with anyone for that matter) review what you are doing. Answer the following questions:

What is my goal?
What is it going to take to achieve that?
What are the action items and by whom?
What are the critical dates and items?
What could go wrong?
What are the other alternatives?
Why might those alternatives be better?
What concerns of this executive does this address or not address?

Come to meetings and presentations prepared. Do not overwhelm the person with information, but since you are prepared you can carefully craft your answer to have the most positive impact.

People who are prepared earn respect in the work environment. Preparation implies professionalism.

Summarize Your Action Plan

When you have a meeting with an executive to decide something, at the end of the meeting summarize your understanding of the decision and what actions if any will be taken, and by whom, and by when. This ensures that you leave the meeting with a clear understanding of what will happen and the executive will know that you understood. It confirms the decision and focuses on the future actions.

No Surprises

One of the things executives hate the most is to be surprised. Let them know ahead of time if something that will impact them or others will not be done on time. Also do not wait to the last minute to tell them that things are not going well. Besides leaving the executive in a weak position with other executives, it leaves people frustrated, upset, and angry. That is not what you want the executive to remember about you when they are doing your performance review. It is unprofessional to attempt to cover up or hide the fact that a project may not be going well. It is a sign of maturity and professionalism to seek out support and take appropriate actions to ensure the success of a project. Making things work is what counts, not so much how you got there - on your own or with help.

Get To The Point (or Lose The Details)

As I pointed out in the abstract for this paper, the amount of detail a technical manager attempts to communicate to an executive may not be necessary. As a matter of fact, it may be too much. As has been stated earlier, executives are concerned with key issues and decisions. As a technical manager your job is typically to manage the details, surface major issues for the executive and provide an appropriate amount of information for him to make an informed decision, and then to implement that decision successfully.

Respect The Use Of Their Time And They Will Respect You

When you meet with an executive come prepared having thought of potential alternative courses of action and having considered the positive and negative factors for each. In addition, prepare an agenda for group meetings and distribute it prior to the meeting or at the start of the meeting. For one on one sessions I also recommend you develop an informal agenda and bring it to the meeting and if the executive asks for a copy of it, provide it and do it in the future.

Often executives will use the agenda list to jot notes on and for listing follow-up items. I suggest you leave lots of space between agenda items for notes and keep the number of items on the agenda list to less than 7. If you have to cover more than 7 items, list the general categories on the agenda and then include the detail in an additional document, in outline format.

Executives appreciate their staff being organized and using their time well. If you feel stilted using a written agenda in a one on one meeting, prepare your items for discussion in order of importance (high to low) and only use the agenda list at the end of the meeting to prevent forgetting a critical project or item.

Get To The Bottom Line

Executives especially those who are unfamiliar with technology very quickly get bored with too much detail, even if you are sure they need to know it. Get to the bottom line quickly in meetings and presentations. If executives need to know more detail, they will ask for it. Present them with high level summaries and status reports. Have detailed documents available so that if they ask, you can immediately provide them with the full picture.

Executives usually coordinate the activities of several managers. By using their time wisely and being concise, when you do need to meet with them to discuss an important issue or emergency they will understand the priority.

Executives Ask For Details When They Are Uncomfortable With The Recommendation Or Limited Alternatives

When an executive feels constrained into a decision, or is not clear about what actions to take, or is not sure of your decision making abilities, then they will ask for details, more analysis, etc. As you work together, deliver projects on

schedule and on budget, and develop a consistent reputation for quality work, less detail is typically required and more accountability.

Start off with the main points in a meeting or presentation. Providing an appropriate level of detail indicates an understanding of your audience, accurate judgement, and good business sense. It also indicates that the manager understands the business priorities and can adapt to them.

Responding to requests for more information can provide insight as to how the executive analyzes situations and what they are looking for. Provide it.

Tell The Executive What The Meeting Is About And How Long It Will Take

Usually you go through a secretary to schedule a time with the executive. Provide that person with the information necessary for them to set it up. If you have a preference as to the time of the meeting, let the secretary know. If the executive always is running late, you may want to get an early morning appointment so as to not be put off till the next day.

Do you have several minor items to meet about or one critical issues that must be dealt with quickly? Make sure the urgency is clear: either a meeting is required fairly quickly or can be loosely scheduled.

Provide the secretary with a time estimate of how long you think it will take. IF you ask for a 30 minute meeting and get it, keep it to 30 minutes unless the executive wants to keep discussing an item. IF you are finished early, great! Use the remaining time to continue building the relationship or leave early. Develop a reputation for being organized, getting to the point, and not dragging things out.

Make It Easy For Them To See Your Point Clearly

Provide the executive with a brief and clear statement of the issue or project purpose, cost (either estimated to the best of your ability or actual), planned benefits (both hard and soft dollar benefits), who is involved, action plan or recommended solution, and what the other alternatives are, if any. Be prepared to present your logic and any detail backup so that they can make a decision.

Provide the executive with a briefing or summary report so that they have a physical representation to take with them. Make it good - quality of work, analysis, and the presentation itself. State the main points clearly. Learn how to present information so that is maximally received and comprehended (sometimes a combination of pictures, graphs, and words).

Take opportunities to practice speaking about projects and issues so that your speaking flows smoothly and naturally. In addition, you will be able to gather others thoughts and feedback about the item or situation which will make you better prepared for presentations.

Make Sure Of Your Facts
Getting caught with inaccurate or erroneous information destroys ones credibility. Be known as a person who respects the facts. Spend the time to check your facts and gather the different points of view. Watch the tendency to jump in too early and make a hasty decision that you may regret later. On the other hand, do not spend too much time thinking and talking about something, at an appropriate point take action and make things happen.

Do Not Run Your Agenda On Them
Executives like to make their own decisions and do things their own way. Provide them with what they need to do that.

Do not attempt to educate your executive about technology unless they request it. Provide them with the information they request and as needed to address issues and make decisions. Do not go into lengthy detail explaining technical details.

It is better to have an executive who know little about technology who you work well with than an educated executive you have not established a successful working relationship with.

Summary

While technology has advanced at a tremendous rate over the past decade, it seems that people still work in much the same way they did 50 or 100 years ago. We work with people to get things done and the key to success is establishing working relationships and communication.

This paper has presented several tips and techniques for increasing your opportunity for successful working relationships with your executive. These techniques and many others are taught in communication classes, management development programs, and in business schools.

This paper is intended to whet your appetite. To excel in this arena, takes study and practice - the two go hand in hand.

A Reasonable Approach to Software Development

Don Gholson
Innovative Software Solutions, Inc.
10705 Colton Street
Fairfax, VA 22032 U.S.A.
(703) 273-5025

Over the past few years we have heard more and more about the "application backlog" and "user unrest". Many methods have been proposed to solve the problems: structured programming, CASE, object management, 4GLs, etc. We still have the problems and more frustrations. This paper explores some reasonable alternatives that will not totally disrupt the development process.

I do not intend to present any new, brilliant, all-encompassing, super-duper method for the instantaneous development of perfect software. Others can try that. I do plan to discuss the basics of what we do, how we go about doing it, and what we might do to improve the way we do things. I hope that in the process that we can inject some common sense into the way we go about our business.

First, we need to define what it is we are supposed to be doing. We are to provide services, and in some cases goods, to units within our organization in order that they may better accomplish their jobs. By definition, this makes us a staff function. We are not here to build empires, large or small, but to assist others in their tasks. Additionally, we are custodians of certain assets of our organization which we have a duty as professionals not to squander, but to use in the most efficient means possible. These assets are the people, facilities, equipment, etc.

We provide equipment and serve as an information utility to our "customers". I'm not going to pursue that function. What I want us to look at is the development of the software for our information utility.

We have heard over and over about how inefficient we are in developing software. We never take time to design, but spend most of our time "maintaining". In reality, we do the majority of our design work during the maintenance of our software. We build software so that it is just good enough to put into production and then we keep refining it during

the maintenance cycle in order to get it to actually meet the needs of the ultimate users.

Many writers, gurus, academicians, and others have proposed a broad range of methods, systems, and disciplines to solve these problems. We have structured this and structured that. We have computer-assisted this and computer-assisted that. All of them appear to be a substitute for the one thing we need most - lots of common sense.

The approaches which we have seen since the mid-sixties have one thing in common. They all assume that by taking an engineering approach to any problem, you can isolate the one right solution.

There are major problems with this. Computer programming and analysis is only partially based on engineering principles. There is no one, unique correct solution to each programming task. Programming and analysis require both the discipline and adherence to basic rules as does engineering and the creativity and fresh idea processes of the arts.

If the engineering solution were absolutely right, we wouldn't have all of the models of cars we have. We wouldn't have all of the different styles of houses we have. We wouldn't have multiple computer languages!

Software design and development is like designing and building a car - many can do the engineering to build a car that runs, but few can add the artistry that makes the car attractive enough to sell.

Lest you think I am being unduly harsh to the engineering profession, I must admit that I started out as an Aerospace Engineer and pursued that profession until I got involved with computers in the late 1960s.

Let us first analyze what our problem is. It appears to take too much time to develop computer applications. Often, by the time the application is operational, it doesn't meet the user's needs.

Right away common sense gives us a clue as to what is wrong: Our end product is not clear when we start the job. Furthermore, as we pursue this unclear goal, we rarely check to see if there is any additional information available that might let us home in on our target. We haven't properly defined our task. We usually haven't involved the end-user.

We haven't monitored the process along the way to make sure that we are constantly heading toward our ultimate goal.

I can see many of you shudder and shake when I mention the involvement of the end-user. Let's face it - if the customer doesn't like the product, he is not going to buy it. It is our job to meet the information needs of the other members of our organization. We may be smart, but we aren't smart enough to know exactly what the end-user wants and needs (note these are often two very different things and it is our responsibility to ferret out the wants and the needs and make a valiant effort to satisfy both).

How we accomplish this is one of the most difficult tasks we face: we LISTEN to what the user is saying and what he is not saying. We ask questions to draw out information and we LISTEN to the answers. When we ask questions, we must do it in plain English, no technical jargon, no computerese, but plain, office-style English.

A man went into a garden store and told the clerk that he needed a lawn mower. The clerk described the three models that were in stock and urged the customer to select one. The customer was uncertain and after some more discussion he left the store.

A second customer came into the store and a different clerk assisted him. This customer also wanted to buy a lawn mower. The clerk asked him if he had many trees in his yard . The customer wanted to know what that had to do with buying a lawn mower. The clerk explained that if he had trees to mow around that a rear bagging mower would be more appropriate than a side bagger. After further discussion and questions, a specific model was selected and the customer bought it.

The first clerk was an order taker. The second clerk wanted to meet his customer's needs. (Of course we could take this one step further and have management telling the clerks to push a specific model.)

The questions we must ask are:

- Do we know what the customer wants and needs?
- Do we have all of the resources to translate these needs into an operational computer system?
- Do we have any idea how well we are doing?

It really does not make a whole lot of difference which method you choose to develop the software. There are plenty of books and seminars out there to teach you how to use them. You have to pick the method that best fits your organization's needs and its culture. The key to it all is to use some common sense, do the planning up front, listen to the customer, and constantly evaluate how well you are keeping on track.

William E. Perry, in his article "A New Kind of Software Virus: Analysis Paralysis", wrote:

> John Toellner, president of Spectrum International, often has said that companies normally spend more money deciding what system development methodology to acquire than what the methodology actually costs.
>
> If he is right, it actually might be cheaper to make a quick decision and get the product in and try it.
>
> If we're wrong, we could throw it out and bring in another one, and it still wouldn't cost us any more than the funds expended through analysis paralysis.
>
> Analysis paralysis is bad for three reasons.
>
> First, it delays the introduction of a product that might improve the quality and productivity of our organization.
>
> Second, it sends a message that we in information systems don't know what we're doing.
>
> Third, it is frustrating to the individuals working on the team.

Doing the planning up front gives us a clearer, more accurate picture of our goal. It also reduces the time it takes to do the coding, testing, training, and implementation. If the user is closely involved, mistakes and misconceptions can be corrected before they become costly mistakes. A problem at the coding stage costs roughly ten times as much to correct than at the planning stage. A problem at the implementation stage costs roughly ten times as much to correct as at the

coding stage. The point is to correct the problems as early as possible and to avoid them if at all possible.

Another key element is to keep track of how things are going so that you can foresee problems and handle them before they become unmanageable. To do this, acquire a project management package and run it on your mini or micro. Make sure it has good graphics and can handle your project (size, complexity, etc.). Then, remember that you drive the software, it shouldn't drive you. The project management package should serve as your electronic notepad. You make the decisions. Don't use the argument "the computer said ... ".

Break the project down into sufficient detail that you are able to accomplish some milestone each week. That way you can have the pleasure of completing something each week and "marking it off the to-do list". This also means that if you start to deviate from the schedule you have set, you can immediately correct the problem instead of waiting weeks or months to find that you are off-target.

Keep the users involved, that way they "buy in" to the project and they are kept constantly aware of the progress and problems. This will also reduce training and implementation time and costs. Use the graphics to present information to users and management.

If one is to believe the trade papers, every data processing shop has at least a two-year backlog of projects and there are dozens of large and small projects in the backlog. That sounds like a situation that is just ripe for some experimentation. What have you got to lose? We can't be doing any worse than we are doing now.

Take one of the smaller tasks. Develop it using new, or at least better, techniques. You don't have to try every thing at once. There are plenty of projects to go around. Try improving the planning up front. Try modelling the software. Try prototyping the system and then go ahead and develop it in your traditional method. On the next project, go one step further and use a fourth generation language to develop the project. At the same time refine the planning and modelling based on what you learned from the first project.

If you have a large enough staff, you might have two experimental development projects going at the same time, using different techniques. Have the teams meet regularly to discuss what they have learned, both good and bad. Foster

competition between the teams, but make sure that it doesn't get out of hand.

Most shops can not afford to suspend everything for a few months (or more) to do all of the tasks necessary to establish a system dictionary. This has to be attacked in an incremental approach. Can you imagine the airlines in the late 1950's and early 1960's telling the public, "Sorry folks. We're going to shut down operations for a year while we make the transition from props to jets. When we have everyone trained, we're going to scrap our entire fleet of prop planes and replace them all with jets. See you next year."

A simpler method that could produce similar results is to take "partial" advantage of CASE techniques - prototype in a 4GL and then when the design is refined, recode what is performance critical in COBOL. Why COBOL? Because about three-fourths of all business applications are in COBOL. There is a wealth of expertise available, it is easier to find COBOL programmers, and the language is more transportable.

For example, I was asked to develop an inventory management system to keep track of all of the hardware in a large network of HP-3000s. I developed a prototype in Transact. Transact was chosen because it gave me an opportunity to learn a new language, I could code it very fast, and I could make changes rapidly to refine the model. I showed the prototype to the users and we refined it to do the tasks they wanted. When they finally decided to cancel the task because they were unwilling to invest the manpower in collecting and converting all of their data to machine readable form, I had not made as major an investment as would have been necessary if the prototyping had been done originally in COBOL.

The trend today is for slow - very slow - movement to CASE. Why? It requires a major investment in time, something that most shops don't have.

Which is more important - being the most "leading edge", technically sophisticated shop or solving the user's needs in an effective and efficient manner. Many sites are on real ego trips and have forgotten that their primary purpose is to provide cost effective, efficient services to their users.

Beware of "methodologies" that are rigid - a form for everything and everything on a form. Look at how innovations fit into your culture. Look at how your culture can evolve to encompass more productive methods. As long as people are

happy and content with what they are doing, they will not see any need for change and will resist it. Change can only take place when a need for change is perceived.

We often go overboard in formalizing methods and in so doing, we preclude solving many problems.

In a recent issue of Datamation was an article titled "That Fuzzy Feeling". It discussed a form of logic called fuzzy logic. It sounds funny, but its been with us for a long time. It really describes the way we think. Trying to force humans (programmers, analysts, users) to think like machines in formal, discrete solution methods severely limits their ability to solve problems. It limits their choice of solutions. Instead of jumping into a rigid logic system, start by thinking fuzzy (build a prototype). As the solution is refined, the thinking will become less fuzzy and thus easier to adapt to our "less than fuzzy" computers and languages.

Don't be drawn into the latest technological fad just to be the "leader of the pack". You may end up like the hare in the tortoise and hare fable - fast but last.

A good example of this phenomenum is the current rush by the Air Force to build the B-2 Bomber. It is a very high tech solution to a moderately low tech problem. The B-2 is a subsonic bomber designed to evade enemy radar. The same problem was encountered in World War II. The British solved the problem by building the Mosquito bomber. It had a wooden skin to absorb radar. I'm not proposing going back to wooden skinned aircraft, but a half a billion for a single copy of an aircraft is a little much. This is a prime example of being blinded by the desire to use the "highest" technology when a less elegant solution might be more efficient and effective.

Train your staff in stages. Don't go "cold turkey" into any situation. Take a conservative approach. Train a nucleus across a range of programmer's capabilities. Mix the trained nucleus with the remaining staff when prototyping and converting in COBOL (or whatever 3GL).

Don't try the "grand design" approach - break tasks into manageable pieces. Programming teams should have no more than five team members - otherwise it is impossible to coordinate activities and individual members lose there ability to express views. With more than five, one or two of the people will almost completely dominate the team.

Finally, It is possible to improve the software development process. It is not an overnight thing. The changes take time to learn and implement. Remember that you are working with people, not machines. Treat the people in a responsible, professional manner. Keep everyone well informed. Plan. Over time, the results will amaze you.

References

William E. Perry, "A New Kind of Software Virus: Analysis Paralysis", Government Computer News, Vol. 8, No. 21, October 16, 1989.

William E. Perry, "Know Thy Customer for Improved System Design", Government Computer News, Vol. 8, No. 22, October 30, 1989.

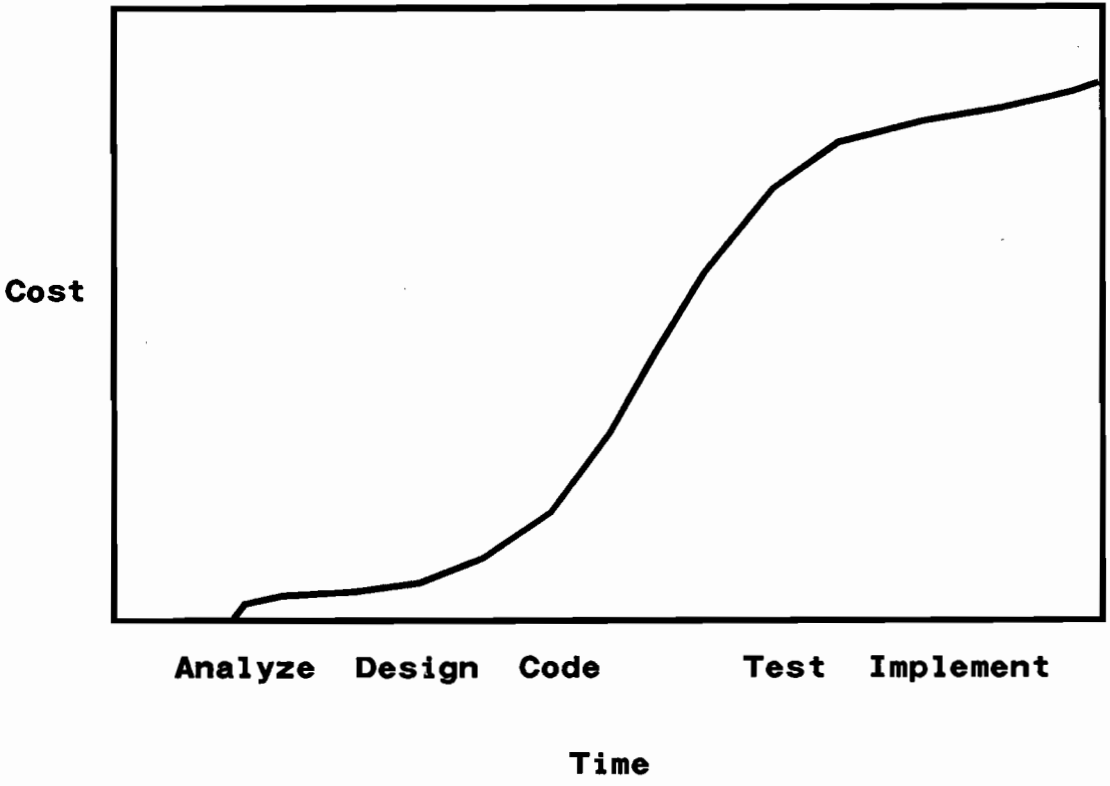
Steven Pfrenzinger, "CASEing the Cobol Joint", ComputerWorld, Vol. XXIII, No. 41, October 9, 1989.

Mark Duncan, "Preparing end users for CASE", ComputerWorld, Vol. XXIII, No. 40, October 2, 1989.

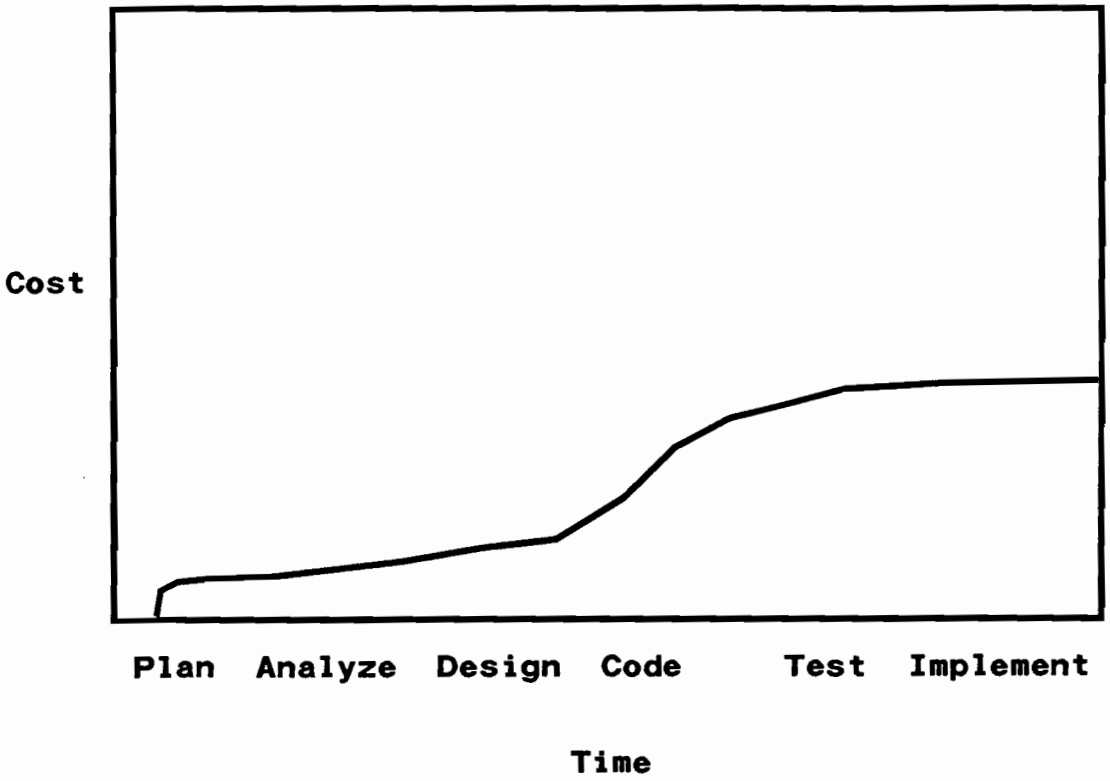
R. Colin Johnson, "That Fuzzy Feeling", Datamation, Vol. 35, No. 14, July 15, 1989.

Earl Hoskins, AT&T, Talk presented on behalf of Index Technology, Arlington, VA, Nov. 2, 1989.

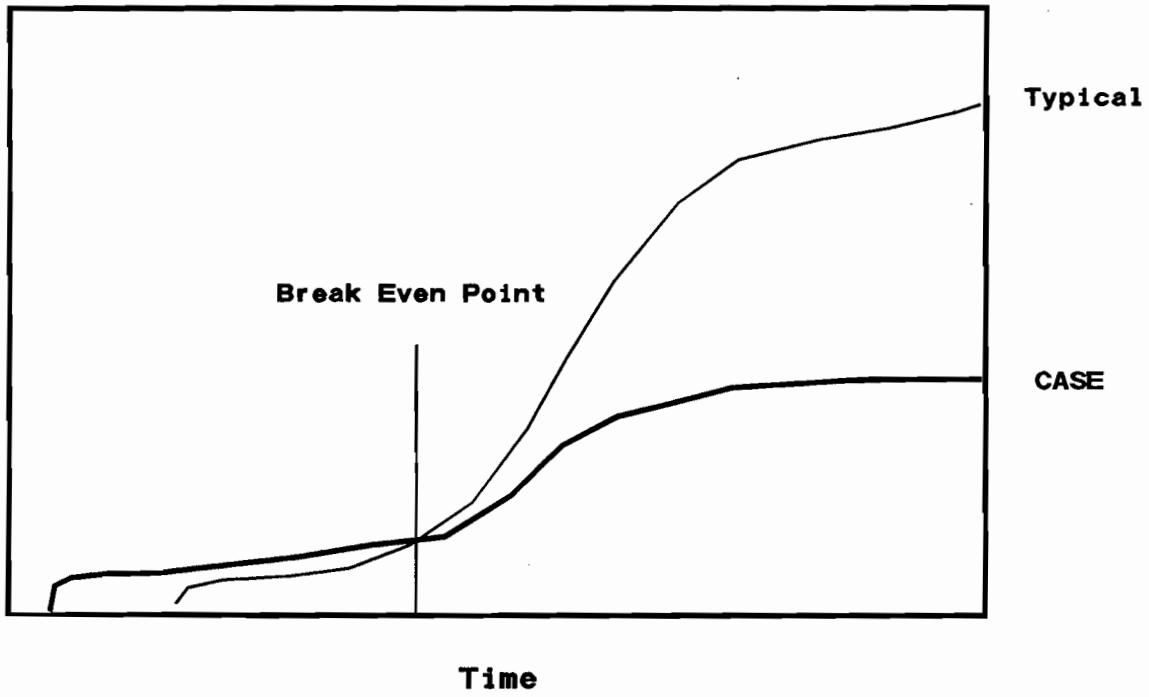
TYPICAL DEVELOPMENT MODEL



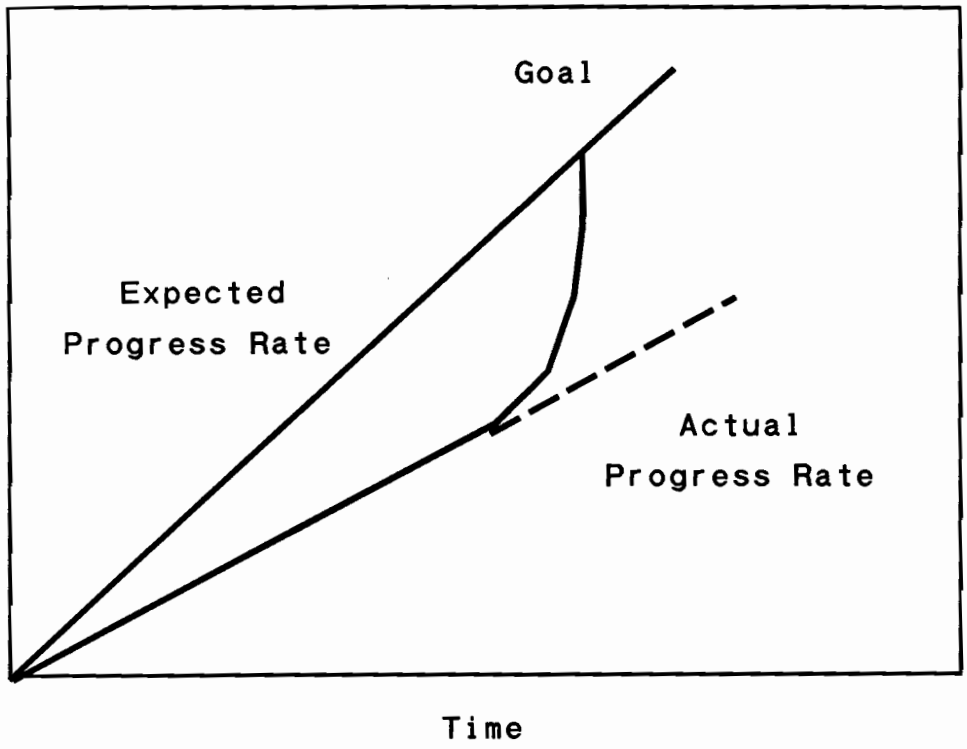
CASE DEVELOPMENT MODEL



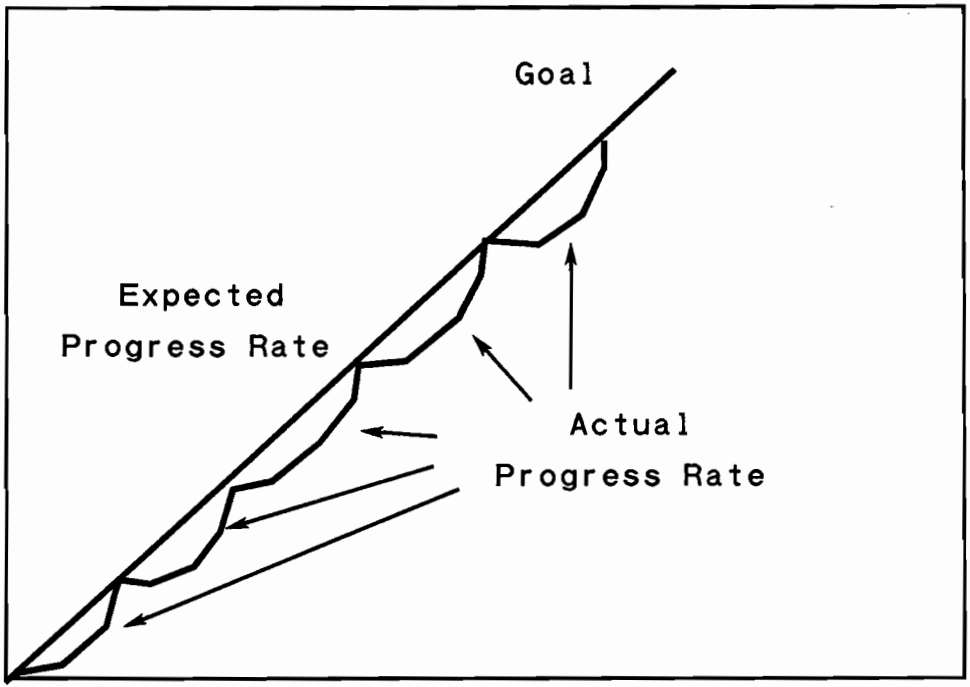
CASE TUNNEL EFFECT



Measuring Milestones



Measuring Milestones



EXECUTING A PRESENTATION

Larry Boyd
M. B. Foster Associates
P. O. Box 851615
Richardson, Texas 75085-1615
(214) 414-0487

INTRODUCTION

Many people, when asked to give a talk or a presentation, immediately go into a panic. Their first thought is the fear of being in front of a group of people. Their second is the fear of being so nervous that the audience will be able to hear their knees shake!

This presentation is to show how easy it really is to execute a presentation or talk. You can either kill the presentation or kill the audience.

STEPS TO AN EFFECTIVE PRESENTATION

The goal in making any presentation is to be able to communicate in an informative and effective way. This can be done by following seven steps:

1. **Know Yourself.** This means that you must be aware of your limitations as well as your capabilities. You must be comfortable with the subject, and your style.

2. **Know Your Audience.** If possible familiarize yourself with their backgrounds, education, occupations, reasons for attending, etc. Always consider your audience throughout the talk. Remember that they are there to learn from you and have rights as listeners:

   a. The right to a message worth the time and effort they have invested in listening.
   b. The right to an orderly presentation of material.
   c. The right to direct communication.
   d. The right to material which is specific, factual and therefore interesting and informative.

3. Know Your Material. The best way to kill a presentation is to not know your material. If you know it well, you will be self-confident, which in turn, will help you overcome stage fright. Also, when you know your material, your audience will feel confident in your credibility.

4. Plan And Prepare - Be Organized. The material used for a presentation should be organized into the three parts which most of us learned in English 101; The Introduction, The Body, and The Conclusion.

   a. Introduction. Use an attention-getter: a story, a rhetorical question, a quote, or a joke. Remember that an attention-getter MUST be pertinent to the talk!
   b. Body. The body should be logically arranged. Use supporting material to backup points such as quotes, definitions, illustrations, etc.
      When you use audio-visual aids remember they are supplements to your material. Do not let them dominate. But remember that an audience will remember more of what they see than what they hear.
   c. Conclusion. This is the last opportunity to achieve your objective, so always carefully repeat your main points. Repetition reinforces memory retention.

5. Know Your Facilities. Physically check out the room in which you'll be speaking when possible. Is it set up the way you requested? Can everyone see the flip chart or A/V screen? Is the sound level okay?

   Sometimes it is not possible to change the temperature of a room, but take that into consideration. If the room is hot your audience will tend to fall asleep or drift off. If it is cold you won't have to worry about them hearing your knees shake because theirs will be! Room temperature can be an excellent attention-getter during an Introduction.

6. Rehearse/Practice. If there is one point I want to leave with you it is this one. This is your chance to iron out any bugs. Is you talk too long? Too short? You will also build your confidence because you will Know Your Material.

   a. Don't Memorize. When you memorize it comes across as "This is a recording" type message. And what happens if you lose your place?

b. Don't Read. You want the audience to at least think you know what your are talking about! This is the opposite of memorizing. The audience will think you have to read because you don't know what you wrote. (Notice the President doesn't write his own materials!)

c. Do Use Notes. Unless you are doing an "off-the-cuff" type presentation, use notes. If you have a podium you may choose to use the orginal paper or 8 1/2 x 11 inch paper. The important thing is to be able to come across in a organized and logical fashion.

   If you know your material, it is organized, and you have rehearsed well, the notes will only be used as a "stack marker" type aid. They simply remind you of where your are. But I don't recommend ever doing a presentation without the notes, no matter how well you know your materials.

Remember, practice is very important for you and your audience. When you have practiced you can learn to maintain eye contact -- a must in a good presentation!

7. **Delivery.** The final point for executing a good presentation is the delivery. You should have no problem, if you followed the first six steps! Just Relax, Breath!, and don't worry about your hair.

When you use your hands, don't fidget with a pencil, marker, pointer, etc.

If you are comfortable using gestures with your hands, do so. If not, simply let your hands rest at your sides.

Maintain an enthusiasm about your topic and a positive attitude toward your audience.

Keep a conversational tone in your voice. Do not lecture and most importantly, do not "talk down" to your audience. Talk with them!

Above all, **BE YOURSELF.**

HOW TO OVERCOME STAGE FRIGHT

Here's some ideas that I have been taught and have picked up by watching others.

1. Be confident you are well prepared and rehearsed.

2. Believe you can do it. Do not let the anxiety defeat you.

3. Before you start, take deep breaths and use total relaxation exercises if you feel overwhelmed.

4. Remind yourself that you are well prepared. You are nervous, but you are also prepared.

5. Winston Churchill overcame an early fear of speaking by looking at his audience and imagining they were sitting there without clothes on!

6. A psychologists offers this advise. A good approach to take is that stage fright never really goes away -- you learn to live with it and use it to your advantage.

7. Act authoritative and positive, no matter how shaky you feel. Keep in mind that people do not know you made an error unless you let them know.

8. Resist urges to play with microphones, pens, etc.

SPEAKING HINTS

Here's a couple of speaking hints:

1. If possible, do not sit while you are speaking. Your are the presenter and you are in control.

2. Be very aware of using "non-words" (er, um, uh). Practice will help this.

3. Avoid lecturing. As we talked about earlier, don't lecture. Talk with your audience.

4. Allow for variations or last minute changes. Don't be so set in you presentation that any change will cause you heartache.

POSTURE

Stand tall, shoulders up, chin parallel to the floor, feet comfortably apart. Do not slump or lean. Think of all the things you mother told you when you were a teenager!

BAD STAGE MANNERISM

How about a list of categories of speakers.

1. **The Caged Lions.** They walk in measured paces back and forth on the stage.

2. **Pinball Machine Players.** They grasps the podium, leaning it, or themselves, one way then the next. An easy way to "Tilt".

3. **The One-Man Band.** Coins or Keys in pocket.

4. **Teeter-Totter.** Rocks back and forth, heel to toe.

5. **Jack-in-the-Box.** Raises up on tiptoes constantly.

6. **Chalk-chucker.** Tosses chalk in air and catches it. Tosses it on to chalk shelf. Could also use markers.

7. **Hitch-Hiker.** Hitching up his pants with his forearms.

8. **Face Rubber.** Seems bored. Continually rubs eyes and forehead with studious concentration. (Variations: Nose-nudger, Mount-mauler, Ear-puller, Cuticle-cutie)

9. **Fig-leafer.** Holds hands clasped in front. (Variation: Reverse fig-leafer -- hands in back).

10. **Dresser-upper.** Fiddles with clothes and hair.

11. **Dishwasher.** Pulls on shirt sleeves.

GESTURES

1. Hand movement must be coordinated with the verbal material.

2. Use variety. Psychologists point out that attention is rarely fixed for more than a few seconds. Unless change occurs, attention moves toward inward thoughts or external conditions. Variety of gesture and movement help create attentiveness.

3. Let hands and body fit the occasion and the audience. (Small audience and routine ideas equal smaller gestures. Large audiences and special ideas equal expanded gestures).

4. Types of gestures.
   a. Enumerative. Number of fingers.
   b. Descriptive. "this high", "this big".
   c. Locative. "behind me", "over there".

WORDS

1. Articulation. "Jeet"-"Jet". "Did you eat Jet?"

2. Pronunciation. Know it before you use it.

3. Vocabulary. Use good vocabulary. Do not use "Know what I mean?" "See?".

4. Appropriate Words. "Never underestimate the intelligence of your audience: and never overestimate its information." Raymond Clapper. Watch those TLAs (Three Letter Acronyms)!

5. Imagery Words. Use imagery words.
   a. Sound - Silence, ring, hum, tick-tock, pop, drone.
   b. Sight - Shape, color, movement.
   c. Taste - Tangy, sweet, sour.
   d. Smell - Sweet, rotten, honeysuckle.
   e. Touch - Rough, smoothy, silky.
   f. Temperature - Hot, cold, cool.

CONCULSION

As you can see, executing a presentation is not as difficult as you may once have thought. These ideas and steps can make a small cost justification meeting to a large technical INTEREX presentation easier, more enjoyable for you the speaker, and more enjoyable for your audience.

Be yourself and enjoy!

MANAGING SOFTWARE QUALITY

Richard Harter

Software Maintenance and Development Systems, Inc.
96 Commonwealth Ave.
Concord, MA 01752

(508) 369-7398

Abstract

This paper discusses software quality and how to manage for software quality. In the final analysis, software is produced by people. Software quality depends on those people, how they work, how they communicate, and how they are managed. The focus of the paper is on these issues.

INTRODUCTION

This paper is about software quality. It is about managing quality. And it is about the importance of feedback and communication in managing quality. My goal is to discuss some of the general management issues surrounding quality rather than to discuss particular tools or techniques.

There is an old saying "To err is human, to forgive is divine." From this we might suppose that people rate pretty low in the quality department. We seem to think so. We are always using phrases like "human error", "nobody's perfect", and "we're only human."

But consider a human being from a systems engineering standpoint. A human being is a real time system consisting of trillions of component processors called cells. It operates in a wide variety of environments for periods of up to a century without fatal system crashes. If we are honest about it, we have to admit that we, the creators of complex software systems, create systems that are many orders of magnitude less effective, less complex, and of lesser quality.

This tells us something. It tells us that it is possible to create systems with much higher quality and much greater complexity than the software systems that we are creating today. But what is quality and how do we get it?

Let us dispose of a superstition. There is no such thing as absolute quality. We, as human beings, are miracles of biological engineering. But we, as human beings, those same miracles of advanced systems engineering, are decidedly imperfect. We all make mistakes. Our bodies have design faults and system failures.

Moreover, the meaning of quality depends on the situation. The greatest carnivore of all time was Tyrannosaurus Rex. He came with six inch steak knives for teeth, eight tons of ferocity, and a gullet that could swallow a human being in one bite. Sure T. Rex was the epitome of some sort of quality. Or was he? How good was he at piloting a spacecraft? Could he write a compiler? Was he a world class skier? Of course not. T. Rex was good at some things and a total washout at others. Just as you and I are and just as our software systems are.

We, as the creators of complex software systems, have to admit our limitations. We don't know how to create systems as complex and effective as the human body or even the living cell. A bit of humility is in order. Perhaps we should start with "Can we do better quality software and how?"

WHO IS RESPONSIBLE FOR QUALITY?

So, who is responsible for ensuring the quality of software products? Is it management? Management, after all, has the ultimate responsibility for what goes out the door. Management dictates what resources are available, the policies that are followed, and who is hired and fired.

Or is the "we" who needs to consider "doing better" the software engineers, the technical people who design and build the software? They, after all, are the people who do the work. They are the people with specialized knowledge. They are paid to do it right.

Or perhaps "doing it better" is the task of dedicated QA specialists? Management lacks the technical knowledge; software engineers lack the control of resources and perspective. Maybe we need people with the power of management and the knowledge of the software engineers to oversee the process to ensure quality. Maybe so, but probably not.

One answer is that quality is everybody's responsibility. That doesn't go far enough and smacks of an easy answer. The problem, as they say with Socialism is that when something belongs to everybody, it doesn't belong to anybody. The same is true with responsibility. If everybody is responsible in a general way, nobody is responsible in particular. The real answer is that everybody has specific responsibilities that are their own, but they must all cooperate with everyone else if there is to be a quality software product.

WHAT IS SOFTWARE QUALITY, ANYWAY?

What is the measurement of quality in software? Is it portability? Ease of maintenance? Ease of enhancement? Ease of use? Speed? Customer satisfaction? Profitability? Long life time? Professional satisfaction? The answer depends on circumstances, the nature of the system, and who you are. This may seem like a fuzzy philosophical point, but until you know what measurements are important to your situation, you can't address quality issues in a meaningful way.

APPROACHES TO QUALITY

There are three popular answers to improving quality. (1) Hire good people, (2) build a QA department, and (3) issue directives from the top demanding more quality.

SKILLED ARTISANS

It's obvious enough that we will have better quality software systems if we hire skilled professionals with a commitment to quality (in whatever terms we've defined it). There is always a shortage of such people, but let us suppose that we have them. Then what?

A skilled software professional is an artisan. Their dedication is to their profession. Our hope is that their professional commitment will lead to quality. And it does. However, that quality is curiously ragged. And the software has problems.

Quality in a software system is, as we have said, many things. No single individual has a complete grasp of all of the criteria, the issues, and how to deal with them. So we build a team of such people whose skills and

knowledge overlap and complement each other. But the software still has problems.

Different people have different areas of expertise and different interests. For example, one person may be interested in storage allocators. As a result your system ends up with a magnificent storage management scheme. Another is interested in dispatch schedulers. So your system gets a first rate dispatch scheduler. The result is that the quality is spotty. It is excellent in the areas where your people have specific interests, but there are shadowy areas which fall through the cracks. This is not to say that people ignore these areas, but they receive less attention because they are professionally less interesting.

Another problem with artisans is that they lack perspective. They aren't end users. Either the developers, the artisans, are users of the product they create or they are not. If they are users, they tend to load the system with esoterica at the user interface because they understand the system very well. If they are not users, the quality suffers because they aren't involved. In either case the developers lack perspective; they don't understand the needs of the end user.

A third problem with artisans is that of coordination. Each artisan does their own thing very well. But it is not enough to have a bunch of good pieces. The pieces must fit together well. Moreover, it is not sufficient to have another specialist put the pieces together because that person has to understand the individual pieces well. We really need to make sure that everybody talks to everybody else, and that everybody ultimately takes care of everything. Which is to say we have issues around the availability of information and around completion of activities.

THE QA DEPARTMENT

A QA department helps. Unfortunately the QA department is generally tacked on toward the end of the development process. The developers complete the software and turn it over to QA. QA runs it through a series of tests and compliance checks. When the software fails the tests, it is sent back to development for rework or a quick fix is rigged and the software is shipped if pressures are too great.

The result is that the QA department is overworked, underpaid, and placed in a politically impossible position. QA is supposed to add, or at least confirm, quality at the end of the development process, which is the most difficult and most expensive place to do it. QA is caught between the developers, who don't want interference with their efforts, and marketing and sales who want the product out the door...now. QA has no natural constituency. Moreover, everybody knows this, so QA tends to become a neglected step child.

DIRECTIVES FROM THE TOP

Another popular approach to software quality is to issue directives. This is popular because it is easy. It is notoriously unsuccessful. A poster urging people to do better is no way to influence software quality. These are, after all, professionals, and they are doing the best they can under the circumstances. There are things management can do to promote quality and we'll look at some of these.

IMPROVING QUALITY

So how do we improve software quality? The many answers fall naturally into two categories: (1) how do we do better as individuals and (2) how do we do better as an organization?

For individuals, the answers are things like upgrading skills, commitment to doing the best job possible, and working with others in a spirit of cooperation. The organization can assist individuals by providing opportunities to upgrade skills by providing training, by providing the tools to do a better job, and by systematically promoting policies that expect and support quality.

Let me give a personal example. A while back I learned that a program I had written a long time ago was being used in a programming course as an example of superior

software documentation. This was quite gratifying. However, the credit for that program and its documentation does not belong to me. Before I wrote it I had worked for a company that had a strict policy that all software would be documented in a thorough and standardized form. Programs were not accepted as complete unless they were correctly documented. I had acquired, as a matter of course, the habit of thoroughly documenting programs. So the credit for that model program goes to the managers who expected and demanded quality as a normal thing.

There is an important lesson here. Most people do not naturally "do things right" as a matter of course. When "right" has so many dimensions, how could they? As the tune from South Pacific goes, "You've got to be carefully taught." It is the responsibility of management to see that people are carefully taught to ask for and expect quality.

Things like good documentation and clean code are good workmanship. There is more to quality than good workmanship. We have all seen instances where people have created unneeded and unwanted software with meticulous attention to detail. But good workmanship is an integral part of quality. And neither quality nor good workmanship will long survive in the face of management that is indifferent to quality, or that systematically sacrifices workmanship in the name of expediency.

FEEDBACK AND COMMUNICATION

The keys to quality improvement are feedback and communication. Many software professionals devote a fair amount of thought and effort to improving their skills and the quality of the software they create. They work at becoming better artisans. This is good, both for them personally and for the organizations they serve. Sloppy and unskilled workmanship is fatal to quality.

But, this is not enough. Individual excellence is not enough when software is created through group efforts. Establishing and maintaining channels of communication and feedback are the responsibility of both line management and of the individual developers. In any organization there are both formal and informal channels of communication. In an organization that supports software quality, relevant channels, both formal and informal, are present.

I emphasize the importance of these channels of communication precisely because their necessity is so easily overlooked. In the press of deadlines and the throes of the creative process it is all too easy to "stop talking" and concentrate on the immediate tasks at hand. This is particularly true when the chips are down, when the first 99% of the job is done and the last 10% remains. It is at that time that the formal mechanisms, the code reviews, the systematic checkoffs and change control procedures tend to be sacrificed. It is at that time that the existence of good channels of communication is particularly important. If you aren't communicating well before the crunch, you won't have time to learn when it hits. When emergencies come, as they inevitably do, it is the informal channels of communication, the pre-existence of established patterns of feedback and information transfer, that will save you.

Informal channels of communication are necessary because they are direct and immediate. They are important. It would be a mistake to assume that the necessary informal channels will develop unaided. To encourage the necessary informal channels there are a broad range of management alternatives, such as encouraging social and athletic groups, establishing work committees, introducing individuals across normal organizational lines who should know each other, and by knowing who your various gatekeepers are and supporting that function.

The function of formal communication channels is to ensure that specific information is communicated to everyone who should have it. Formal channels are easier to implement than informal ones.

Many software management techniques are designed to provide communication channels. Code reviews and design reviews are good examples. A code review session accomplishes several things besides exposing errors. It tells everyone involved what the code is doing and what the person writing the code is doing. A code review makes explicit the assumptions made in writing the code. It also makes the participants aware of their different coding techniques. In short, it is a learning experience.

We have been talking about communication internal to the development group. Other types of communication are also important. As an example, one of the issues in quality is customer satisfaction. What does the customer want? What does the customer need? What makes the customer happy or unhappy?

How do you answer those questions, what information do you provide to the development team, and how do you provide it? One approach is a software problem report system. Such a system serves two functions. (1) Internally, a problem report system ensures that software problems are followed up and acted upon. (2) It also provides a channel of communication for the customer to the developers who are shielded (more or less) from customer contact.

Software problem report systems are a good way for customers to communicate dissatisfaction with specific facets of the software. They are not so good for telling the developers what the customer needs and wants. For high quality, the development team still needs to understand these needs and wants. In most organizations this is the task of sales and marketing. How often do we think of sales and marketing as an integral part of software quality? To some people the idea would be surprising, which tells you something about that channel of communication.

TOOLS AND PROCEDURES

A popular approach to quality is to think that we would finally have quality if only we had the right tools, the right techniques, and the right procedures.

Here are some examples of tools and procedures: Software problem report tracking; configuration management; automated test analysis; automated software builds; automated software metric analysis and reporting.

What these have in common is that they automate and systematize the administrative procedures needed for software management. These tools serve many functions. They take care of administrative functions faster and more accurately than people could. They ensure completeness of effort. They provide channels of communications. And they provide information that would otherwise not be available because nobody could have gathered the relevant data.

Again, I want to emphasize the importance of communication. As another personal example, our firm markets a change control and configuration management system. There are a lot of good reasons to employ such a system, of course, but one important reason is that it lets you describe changes to the software before you make them. This is a formal communication channel. You have a forum for telling others in the organization what is going to happen before it happens, instead of afterwards if at all. Even if you are the only person involved, you are making your plans clear before you implement them, rather than afterwards. This is known as looking before you leap and it is generally a good idea.

Another function of tools and procedures is that they simplify life. They remove complexity and routine decisions. Complexity is not bad in and of itself. The problem is that unnecessary complexity is a waste of resources. Tools can eliminate unnecessary complexity. They let us concentrate on the tasks that are truly hard, that require creative thought, without wasting it on the easy but routine tasks.

TECHNICAL MANAGEMENT

People often ask at this point if is it more important to be concerned about a manager's people skills or their technical skills. The truth of the matter is, of course, that both are important, but that there is a third, and quite distinct, issue here: technical management.

Technical management involves organized thinking about the design, implementation, and organizational procedures. There is a body of knowledge about engineering which is process specific to the product under development and generic to production and engineering. We should not mistake good people skills or software engineering knowledge for technical management knowledge. They are not the same things.

Often technologists wind up making technical management decisions by default. Such decisions are typically well intentioned but uncoordinated since they are made by a variety of people as required. Technical management pulls together the technical and organizational aspects of a project. We need good technical management to have high software quality. In recent years engineering institutions have offered explicit courses in technical management. These, coupled with experience and support from upper management, provide our best hope for the successfully managing technical projects.

CONCLUSION

Tools and procedures are a means to an end. They will not, by themselves, ensure software quality. In the final analysis, it is still people who do the work and people who determine the quality of the final product. We need to be aware of the factors that influence quality of work. And we need to act on that understanding

Software quality management is largely people management of a particular type. It is the type of management that looks at what people do, how they work, what resources they need for their work, where the weak points in their work patterns are, what communication channels are needed, and sees to it that the support and structure needed is there.

CUSTOM OR PACKAGED SOFTWARE

Jean A. Hills, Vice-President
N. A. Hills Computing Services Limited
336 Piccadilly Street,
London, ON Canada N6A 1S7
(519) 672-1731

Are you shopping for a software package that will do all your bookeeping? Have you thoroughly documented your needs before you even look at the software? Are you looking for a package that is cheap or are you more concerned with how the package might fit into your management style? Do you know what your needs are or do you rely on the the recommendations of friends or salespersons?

Buying a software package is like hiring a professional to come into your office. If you were hiring an accountant to handle the financial transactions of your company then you would carefully interview many candidates. You might even hire an employment service to screen applicants for you. You would seriously consider the candidates certification and references. You would scrutinize the candidates past employment records. It would be important to select a person whose work style and personality would promote trust and cooperation with the other employees. It would be equally important to spend time with the candidate outlining the job responsibilities and what your expectations are for this position.

Managers seem to follow all these procedures and more when they are hiring a person, but when it comes to software packages, they are easily influenced by cost and sales promotion.

Buying software is similar to hiring an employee. The manager should first list all the possible items wanted in the package and should be aware of the work style of the other employees. Just as an arrogant, abrasive person can disrupt the whole office, so can a software package that dictates a change in work style. Any new software additions should be as carefully chosen as a new employee.

After you have made your decision and hired a new employee, it is important to take time to answer questions and help the person feel at home in the organization. Just providing a new employee with a desk and chair is not enough to guarantee good results. The physical things are necessary but human direction, and encouragement are imperative to success.

The same is true with computer software. Regardless of the promotional advertising, one should never be so naive to accept a package at face value. Just providing the software with a computer is not conducive to success. It takes time to understand and read the accompanying manual

exactly what the program will do before you let it loose on you office records. The software is a new employee and will perform best when it is carefully selected, understood and installed. Like a new employee, it must be introduced to the staff and be friendly enough to be accepted by the staff. It must be remembered that a package software was created to perform specific operations and it is not always compatible with the expectations of the management and staff.

Some people buy software as casually as they would buy clothes. If the advertisement is convincing or the salesperson is persuasive then that package is for them. Like clothes, software packages reflect the style of the creator. This does not always compliment the style of the user/wearer. Like the very chic dress that hangs in my closet, the software might reflect all the elements of current style but it is not comfortable to the user/wearer. It might generate yards of of reports which are unintelligible to the user and so are garbage, and it could confuse the work currently being done by the office staff because it is not their style. What use is an elaborate budget system to a small operation that operates solely on the present actual.

In my position as sales manager, I have visited hundreds of small businesses and queried them on how they used their computer and what programs they use most.

First and most important was the word processor. For many offices this is the sole use of their computer. I agree that I would be lost without the word processor program, the spelling correction software the spacing and paging software, but to limit the computer to that, would be utilizing only a small part of the computer's talents.

Rather than buy an expensive package, you should explore more powerful ways to better utilize your editing resources. There are good word processing and spelling programs on the market but none of these are cost competitive with the EDIT/3000 and GALLEY that already exist on your system.

Spelling programs generally check the text after it is completed, We have adapted the EDIT3000 to check the spelling of each line as it is entered.

To initially enter the large list of everyday words into our program, we had the help of our eight year old granddaughter. For a penny a word, she earned a magnificent salary and improved her own spelling.

I encountered many bookeeping packages which managers loyally defended with little reference to the reduction of time and cost which these packages claimed. It did not seem to matter how many reports they generated they just scrapped the ones that were of no value. Some did complain of the

enormous waste of paper. Some were so confused by the reports that they put the computer aside and went back to the manual method. These computers often sat dejectedly on a table in the back of the office. One was even offered to me if I would just take it away.

The most pathetic situation was that of a small hose and nozzle company who had bought a software package to do their billing and inventory. Because of the multitude of small pieces in their inventory, it was impossible to put them all on one floppy disc. The operator had to be continually changing discs to complete an order. This was confusing to her and took twice the time it originally took her when she relied on her experience and her memory.

Software purchase and management can be as complicated as the selection of a new employee. Psychological testing would be ridiculous in the selection of software but the purchaser should be aware of the psychology of work style. To impose a new work style on dedicated employees can be disasterous. This work style will be crucial to the success of the program and will not be found in packaged software. A manager's style will vary widely. Often the individual is unaware of these variences in style until forced to conform to the restrictions of a particular program. Just as clothes bought on impluse will often hang in the closet for years, because they do not suit your lifestyle. It is a waste of space and money when it does not compliment your ways. So packaged software will sit on the shelf and the manager will carry on without the use of computer. As I discovered in my visits, many computers sat dead in a corner waiting to be resurrected by a knowledgeable operator and suitable program.

Excuses are always the same. "When I get time to it, I'll sit down and figure out how to use it. Just haven't the time just now." and of course there never will be time.

Personelle selection involves time and understanding. Time to interview suitable candidates and time to elucidate the functions of the job, and the intelligence to understand your style and the social climate of your business. How will the new employee fit into the organization? Will the employee compliment the team already working or will the employee disrupt the working of the team? A software program improperly chosen can annoy and frustrate staff. These annoyances and frustrations can replace a smooth operation with a hostile, uncooperative team.

How many times have you heard people blame delays, irritations, mistakes on the computer? In reality, the machine never makes a mistake, it just carries on and amplifies the mistake created by its human operator. Just as in The Sorcerer's Apprentice, the apprentice creates more problems by his improper management of the broom. Instead of making life easy for the apprentice by carrying the

water, it became impossible for the apprentice to stop the work. So an inappropriate selection of software will make life unbearable by producing unnecessary functions and a copious flood of reports.

Software packages should be carefully and thoughtfully chosen and if there is any doubt in the user's mind, it would be better to have software created to suit the needs of the user.

We had an interesting challenge while setting up a program for a large steel company. It had to be used by each employee who worked in the yard to ensure its success. IMPOSSIBLE, the management said. Most of the workers had worked all their lives in the yard and had little education. It was a challenge, but one that could be met. We listened and watched the men at work in the yard and then set about to simulate their action in a software program. We even allowed for the large, stiff hands of the handlers on the keyboard by minimizing the amount of key work they would have to do. After many weeks of working with the men, there was no one prouder of the program than the workers themselves because they had helped create the program.

In contrast to this, we prepared a proposal for a large photographic company. When we suggested the importance of listening to the people who would activate the computer, we were rudely told that no subordinate would tell management how to run the business. This manager completely ignored that fact that it is the employee who gets things done and to confuse an employee by forcing that person to conform to a packaged way of working, is like letting the tail wag the dog. An employee who helps design a program will accept computerization more willingly than one who is forced to conform to the constraints of the packaged software. Style is important. It is the work style that contributes to the success of the product. Sometimes this style is the success of the company and to ignore it is often disasterous.

Custom software can get things done and is tailored to suit the style of the employee and the manager. It is this unique compliment of talents that make the company successful and to ignore them will make working conditions unpleasant and often destroy the efficiency that a computerized office can give.

The creation of Custom Software starts with four to five half day seminars over a two month interval, to document a potential user's needs for a software package. It is astounding the amount of change that takes place over this period. The final text from meeting five will contain very little of the text from the first meeting. As the ideas begin to flow, it is normal for them to develop and change. It is exciting to watch this development of the the process.

Many potential users have difficulty visualizing what they want or need, until they can be shown something that they don't like. With package software the realization of its strengths and weaknesses are evident only when the package has been bought and fully paid for. Then it is too late to make changes or return it.

Custom Software has the advantage of prepurchase prototyping and documentation which assures the purchaser of ultimate satisfaction.

Package Software is WYSIWYG what you see is what you get, whereas Custom Software provides for WYGIWYW what you get is what you want.

There is always the customer who wants to wait until prices are lower and packages are exactly what he wants. I visit one annually. He has remainded on the fence for 14 years and I am sure he will stay there forever. Each year, I meet a new bookeeper or accountant who has all the answers. These must be incorrect because each year there is a staff turnover. The latest is that he is in financial difficulty which could have been avoided if he had adopted a custom system years ago.

One of our most valued customers says of his custom software and computer system, "It is not that the computer makes him a better manager but that the system offers him the choices needed to be the best manager."

Custom Software like custom clothes can be expensive. Packages are sometimes cheaper but cheap does not always mean best. A program which is paid for and unused can be more expensive than one made to measure. A program that frustrates the office staff and disrupts the team can deteriorate office morale. A local merchant recently bought an expensive retail program designed for retail stores from south of our border. Up until this was introduced it was a family business and even the employees were treated like family. The installation of the program changed all this. The 60 staff members were trained by an efficient drill master. Nothing could go wrong, but like Murphy's Law everything possible did. The employees refused to use the terminals, few of them could type and the large numbers and descriptions were too much for them. The new billing procedures annoyed the customers because they could not understand the format. The last time I was in the office nothing would balance and everything was in chaos. The training team had left and the employees were ready to leave too. I have not been back to see the sequel of this but I hope they can work it out somehow.

Custom Software could have avoided this. True there are always problems which crop up from time to time. These can be modified and dealt with by an efficient backup team which is at the other end of the telephone line.

Custom Software also allows for changes to be made easily when the whims of government agencies make changes to the tax structure, or tax deductions. With a package, it is well nigh impossible to make these yearly changes unless the program has allowed for these. Many do not.

One good example of the flexiblity of custom software is a sportswear company that we work with continually changing and upgrading the software to accomodate the innovations each season brings to the fashion world. There are new incentives to be added, new schedules of compensation, new costs and materials. These keep changing and only a custom program would accommodate these expansions. Every six months we make enhancements as the business grows and changes.

Another customer boasts that he is still working with the same database that we established for him 12 years ago. He likes to do some programming himself and the program is set so that it will accommodate his work. We are called to help occasionally when he gets in beyond his understanding.

Before purchasing the custom package, this customer was very interested in a software package used and highly praised by another manufacturer in the same industry. Later he was able to buy the assets of this company when it failed.

When buying a package, it is wise to investigate the whole package. This means reading the manual before purchase and asking questions before the package is bought. Many software package operations are here today and gone tomorrow. This leaves a customer without any source of support.

When buying custom software it is important to insist that the code be included in the purchase. In this way any intelligent programmer can make modifications to the program should the need arise. One customer enjoys this advantange because it allows him to modify his program when he wants. We get a frantic call sometimes when he has gone too far but we can usually restore the original system.

From our perspective we see a similarity between the selection and training of people and the choice of operating software for a company. Both human candidates and software candidates must undergo a selection process, sustain a training program and adapt to the office management style.

The computer is not a magic box to be activated by a miraculous program. The computer is an employee, which when properly integrated into the company, will provide a proficient and an economical increase in human effectiveness. It works in excess of a forty hour week without requiring extra compensation. It does exactly what it is programed to do, asks no questions, and never tires of repetitious jobs. A well programmed machine will free the human element for more creative tasks, even designing more

ways to use the computer.

Software is not a static boxed article. It is a changing, adapting growing part of the machine that allows it to satisfy its human operators.

Improving Software Quality

by Robert Green

Robelle Consulting Ltd.
8648 Armstrong Road, R.R. #6
Langley, B.C. V3A 4P9 Canada
Telephone: (604) 888-3666
Fax: (604) 888-7731

I am founder of a small software company that makes tools for the Hewlett-Packard 3000 computers. I also spent seven years working in the HP factory and three years in an end-user MIS department. I have been involved in the development of applications, operating systems, and tools, some of high quality and some of low.

Our software products QEDIT and SUPRTOOL have high reputations for quality in the HP 3000 marketplace, but I suspect that our initial design and code are not much better than other people's. What may be different is the way we evaluate and revise our efforts before official release, and the way we react to error reports after the release. The result is that few users ever experience our programming or design mistakes. When one does, we have systems in place that can correct most of them quickly. That is the topic of this paper.

I have good news and bad news regarding software quality. The bad news is: compared to hardware, software quality is terrible. The good news is: I don't think it is the programmers' fault. The techniques for writing good code are well-known and I won't rehash them here. No doubt there are still more techniques and tools to come that will assist in this difficult task. However, most of the quality problems occur because of mismatches between the programs and the needs of clients, not because of bugs.

Outline

Quality Equals Superior Value to the Client

> **The market value of a product is <u>not</u> an intrinsic value, not a "value in itself", hanging in a vacuum. A free market never loses sight of the question: of value to <u>whom</u>?** [Ayn Rand]

Defining Quality

A good way to start any inquiry is to define your terms. According to my *Little Oxford Dictionary*, **Quality** is a noun meaning "degree of excellence". **Excellence** is defined as "surpassing merit", **Merit** as "goodness", and **Goodness** as "virtue". So what we have here is an ethical issue: **Quality is the relative virtue of a thing, compared to alternatives.**

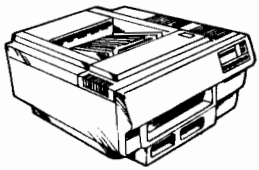
Your software has quality to the extent that it provides **Value** to some living, breathing people with choices and options. If another program solves a similar problem in a way that the person values more, it has higher quality.

Quality Demands Constant Attention

Quality is a hot topic right now with business gadflies like Tom Peters: **"Quality improvement is a never-ending journey.** There is no such thing as a top-quality product or service. All quality is relative. Each day each product or service is getting relatively better or relatively worse, but it never stands still. Ford is doing well now, but Toyota, which Ford sees as its principal competitor, is implementing 5,000 new suggestions a day."

Quality is Different From Correctness

Quality is not the same thing as "Correctness", which is producing a program that exactly implements the design specifications. What if the design does not specify what the clients need and want? Quality is not static: people's needs and situations change over time, and as they do, the quality of your program will change as well. You cannot separate the quality of a product or service from the evaluations of the people who will be using it at a given time and place.



The 150 "worked", but was incompatible with other PC software, used an odd-size diskette, and customers did not want the touch screen. The LaserJet, on the other hand, satisfied customer needs perfectly. [Kathy McKittrick]

Quality is Judged in a Context

Don't ever forget that the quality of your programs will be judged in the context of an installed client system. The client has other programs that he uses and understands, he has administrative policies that may differ from yours, he has people with different backgrounds and training from yours. You must be aware of the client's total experience with your program.

Clients have their own context from which they look at our software:

> **Why do some subsystems require "EXIT", some "E", and some "EX" to exit? Even in MPE XL, in a single subsystem, SYSDIAG, part of it requires "EXI" and the next level up requires "EX". I know that there is an explanation and it is because parts of the software were written by different teams in vastly different geographical locations, but it seems to me that an elementary standard such as exiting the program should be uniform.** [John Dunlop, *Interrupt* magazine]

Involve the Client From the Start

After the state spent $20 million and nearly seven years trying to computerize its public-assistance program, the first caseworkers to use the system made their own discovery: They could figure out a client's benefits faster by hand than with the computer. [*Seattle Times*, May 19, 1989]

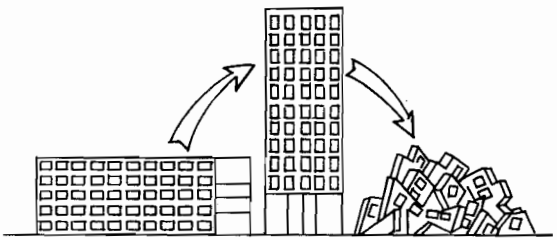
Why Do Big Projects Often Go Wrong?

COSMOS was a gigantic government project to save money calculating eligibility for social assistance such as food stamps and welfare. "After spending over $20 million, the Washington State government decided to swallow its losses and terminate COSMOS. A consultant's report recommended scuttling COSMOS. The report cited poor management, an overly complex design, difficulty to use by caseworkers, and the use of untested software." [*Seattle Times*]

This software disaster includes most of the things that can be done wrong. The state contracted with an outside firm to design and implement the system. What started as a Big Project, grew into a Giant one. State officials bragged that COSMOS would use artificial intelligence. It was seven years before the first pilot installation, when workers found it took up to twice as long to figure out a client's eligibility with COSMOS as it did manually.

The traditional method of developing a DP system includes endless user interviews, voluminous specifications, official approval by confused users, programming phase, integration phase, testing phase, user training, and endless bureaucracy. Notice that this method does not deliver any working programs to the clients until the very end. Creating a DP system without delivering **anything** to the clients until **everything** is done is like constructing a complete office building on its side, then trying to lift it into position.

"TRADITIONAL METHODS APPLIED TO AN OFFICE BUILDING"



Projects Must Be Grounded in Reality

Physicist Richard Feynman was a member of the Presidential Commission that investigated the crash of the Challenger shuttle. He concluded that NASA management exaggerated the reliability of the shuttle to the point of fantasy, then regularly and subtly reduced safety criteria to maintain the published launch schedule.

> **For a successful technology, reality must take precedence over public relations, for Nature cannot be fooled.** [Feynman]

Fantasy by top management has a devastating effect on employees. If your boss commits you to producing a new accounting system in six months that will actually take at least two years, there is no honest way to do your job. Such projects usually appear to be on schedule until the last second, then are delayed, and delayed again. Management's concern often switches from the project itself to covering up the bad publicity about the delays.

> **Information from the bottom which is disagreeable is suppressed by big cheeses and middle managers ... Maybe they don't say explicitly, "Don't tell me," but they discourage communication ... it's a question of whether, when you do tell somebody about some problem, they're <u>delighted</u> to hear about it. If you try once or twice to communicate and get pushed back, pretty soon you decide, "To hell with it."** [Feynman]

An objective project goal unleashes people's minds to discover solutions and attain the goal. An irrational goal just short-circuits the best within them.

Moon's Maxim

Why is it that systems designed with great care, using experienced analysts and the latest design techniques, can totally fail to solve the client's problem? I heard one answer recently:

> **Moon's First Maxim: The process of developing a system uncovers information about the system that <u>no one</u> could have known at the offset.** [Richard Moon]

Users are not, and cannot be expected to be, systems analysts. And systems designers cannot think like users. The client often cannot describe what he wants -- he does not realize how important exceptions are. Even when the analyst extracts

all his wants from him and defines them in an enormous specification, he has no idea what is critical and what is frosting. In an attempt to wrench precise specifications from the client, some shops spend so long on the design that by the time they are done, the client's needs have changed.

The clients are often shut out after the general design phase. They are asked to approve the specifications so that the programmers can get to work. One thing you can be sure of: the clients may not be able to tell you what they want, but they can tell you what they don't like when you finally deliver the code.

The Hewlett-Packard Company has more success producing quality electronic instruments than quality software products. I think the reason is that the engineers who design instruments are basically their own clients -- they can visualize what would make a better product because both they and the client are engineers. The programmers who create financial accounting packages are not accountants, they are programmers -- and they can't afford to trust their personal judgment of an accounting program's quality.

Get the Program to the Client

It was Michel Kohon who first pointed out to me the reason why it is difficult for the client to visualize the result of a program, especially an interactive one:

> **A program is not static. The actions it performs vary dynamically, depending on the information that is entered. It is a moving body and is unlikely to be adequately described without using jargon. The same applies to mathematics or astronomy, or films. How can we visualize a film from a script? This is why the sooner you show the program to the user, the better it will be for his understanding.** [Michel Kohon]

You must get the program into the client's hands in order to find out what you don't know! Once you get a reaction from the client, you can revise the program to meet his exact needs.

Moon's Second Maxim:

Development methodologies that do not support iterative development are doomed to failure.

This is a key insight. You can never get a software design correct by just studying and interviewing. You will always have to go back and revise the design as soon as you start implementing. We use a form of iterative development called the **Step by Step** method that was proposed in a 1980 paper written by the above-mentioned Michel Kohon.

Start Small and Make Constant Improvements

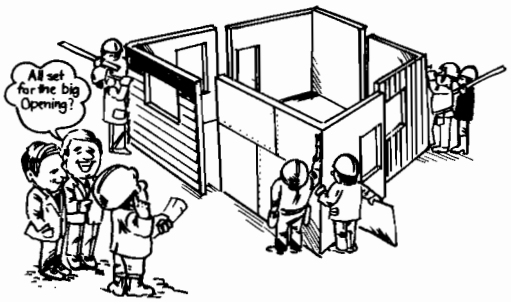
> **American companies also have often lagged behind their overseas competitors in exploiting the potential for continual improvement in the quality and reliability of products and processes. The cumulative effect of successive incremental improvements in and modifications of established products and processes can be very large; it may even outpace efforts to achieve technological breakthroughs.** [*Scientific American*]

The Advantages of Small Projects and Pilots

I have observed that small teams seem to produce quality results more often than large teams. Others have observed the same:

> **At Pacific Bell, a system was required for automating a million transactions. Two estimates were received, one from a big, outside firm (three years, $10 million) and one from a major Pacific Bell unit (two years, $5 million). Meanwhile, three South California employees took a crack at the task--and did it in sixty days for $40,000.** [Tom Peters]

Small projects have the advantage that they can be cut off or modified quickly. Big projects are hard to cancel, because of the political flak over all the money already spent, and are hard to modify, because of the rigid planning that goes into them. To be successful in the free market, you need to respond instantly to new facts and opportunities.



Small projects, especially **pilot projects**, are perfect for testing new ideas in the real world. Even large goals, such as a new aircraft design at Boeing, can be done as a series of small projects. Parts of new aircraft are tried out as redundant systems on current aircraft. According to Tom Peters, the key to successful innovation is to "test it now, at least some piece of it, in the real world..."

> **Is this approach 'soft'? NO! It is hard--the very hardest. It is rational and it is 'scientific'. In fact, it amounts to the organizations embracing the essence of the scientific method--empiricism and the experimental method. Piloting is the approach based on data. By contrast, decision making by proposal churning is whistling in the wind; it is the truly soft and ultimately less rational route.** [Peters]

Step by Step: Limit Time as Well as Staff

Everyone is familiar with Brook's Law:

> **Adding manpower to a late software project makes it later.**

Why is it that increasing the resources never seems to get the work done faster? One reason is economics. To produce programs, you will assign programmers, but there are never enough. Why? Because the client's demands will always increase to match your supply of programmers.

This is a common result in all human interactions. When they opened a new freeway in Vancouver recently, a highway expert said not to expect any lessening of traffic on other routes. The reason: by making it easier to travel downtown, the new freeway would entice more suburban motorists to take trips. The traffic expands to fill the roads available.

> **The only logical way to escape this dilemma is to limit the offer [i.e., supply]. How can we do that? One way is to limit resolutely the number of programmers working on a project. A second way is to limit explicitly the amount of time allocated to a program or system.**
>
> **Let's imagine for a moment that we've said we have two weeks to program our system with the existing manpower. No more than two weeks. How can we best solve the problem in the amount of time given? The natural way will be to put on paper what the MUSTS and the WANTS are. If both can be produced in two weeks, we will program both, but that is unlikely... The most important objective is to find the absolute MUSTS which can be produced with the current staff in a limited period of two weeks....Never go back on the two weeks allowed. It MUST be done in two weeks. Try to imagine that in two weeks' time, it will be the End of the World. Users will laugh, but they will, as well, appreciate your concern.** [Michel Kohon]

The Step by Step method suggests dividing projects into two-week chunks, then delivering each chunk to the client for actual use. This has a number of useful results. It involves the client directly and enthusiastically in the design of the system, it means you never have to write off more than two weeks' work if your design is wrong, it means you can make constant adjustments in your goals as you get realistic feedback, and it eliminates the difference between the development and maintenance programmer. Everyone becomes a maintenance programmer, charged with delivering increasing value to the client in each step.

Continual Improvement

You can learn a great deal about what produces successful software by looking at the HP 3000 software market. The HP factory often appears to follow a method that can be summarized as Design It, Code It, and Forget It. The result is that most of their software products over the last ten years have been unsuccessful. The ones that succeeded, such as COBOL, have had constant enhancements over the years.

The software products that have not had attention after first release are now forgotten. The most successful HP program, MPE, has been continually enhanced for 15 years. But consider the Spooler. The new Native-Mode Spooler is the first enhancement since the SPOOK program, over ten years ago, and the 2680 printer support, over five years ago. No wonder four vendors can make a good living selling enhancements to the Spooler.

If you want to see the tremendous power of continuing, unrelenting, tiny improvements, you just have to look at the Japanese success in manufacturing:

> **The Japanese treat every product as an ongoing experiment and are constantly engaged in improving it.** [Peters]
>
> **Find out what was wrong, try to understand why it had gone wrong, and then break down the corrective process into modest steps.** [David Halberstram, *The Reckoning*]

I feel strongly that a programmer should stay on a program through its life. In many shops, programmers are treated like commodities and shifted from project to project frequently. This ignores the benefits that come from continuing to work on a single project: experience in client needs, experience with the code and data structures, working relationships with the other people involved on the project and in the industry. These are lost if a programmer is shuffled off to another project as soon as he completes part of one project.

Solve the Actual Problem First

Identifying the long-term and short-term objectives will permit you, with the users, to draw a line of actions within an overall strategy. You will move from point A to point Z through points B, C, D, ..., with each point being an objective. But how to order these points? To provide a solution to the top problem [first] means that you will give the maximum result in a minimum of time, and you will repeat this with each successive point. Order the objectives from the maximum payoff to the minimum. These will be your Steps. [Michel Kohon]

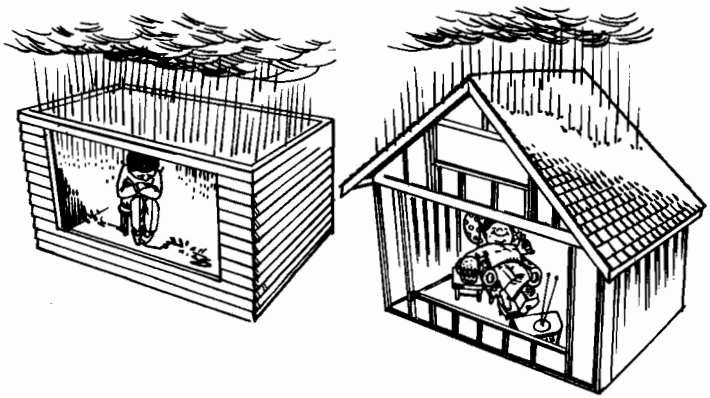
Let the Value to the Client Order the Steps

I have proposed involving the client in the design process, starting small, and improving a system in continual steps. However, what do you work on first, and how do you set priorities?

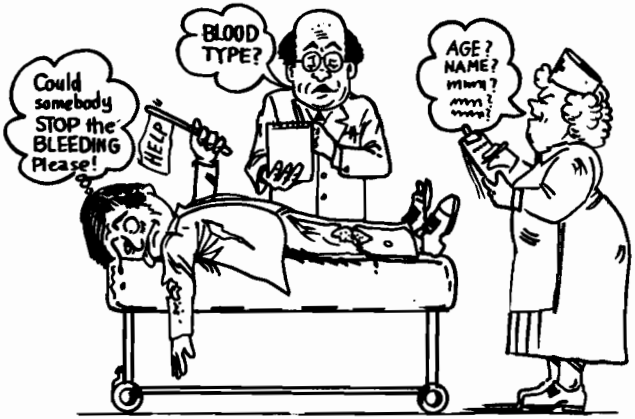
Programmers have a tendency to want to work on the technical challenges first, since that is what they know best. But a beautiful screen doesn't help the client unless it has data on it that are important to him. This is not Step by Step.

Step by Step aims to discover the client's actual requirements and program them all, eventually. Suppose a client is having cash flow problems? He asks you to provide an order processing system, expecting that the more efficient invoicing will bring in cash more quickly.

The typical response is to give him an order p'ocessing system. If you could provide a complete working order processing system in two weeks, including invoicing, you would indeed solve his cash flow problem. But you can't, so you conduct a long study and install order entry as phase one. This is more work for him and does not solve his most pressing problem. This is not Step by Step.



Step by Step challenges you to deliver something in the first step that will make a big contribution toward solving the client's most pressing problem. This is not easy to do -- it takes creative thought. You might automate just the invoices with the largest dollar amount. Or just the simplest ones, leaving the staff free to deal with the ugly invoices manually. Think of solving the 20% of the cases that generate 80% of the benefit.



Solve Part of the Actual Problem in the First Step

Until you deliver a program to the client, you have not accomplished anything, and you haven't started receiving the objective feedback that will ensure a quality system. The advantage of going after the immediate problem first is two-fold: it gets the client on your side, and it uncovers facts about the problem that may make the rest of the project simpler or unneccesary.

This is the hardest part of the Step by Step method, the part that requires the most demanding thought on your part. You will need to analyze the client's problems sufficiently to make an objective hypothesis identifying the most critical problem. For complex clients, this could be a major study. But you must always remember that the goal is the programs, not the investigation.

Know Thy Client

> **Since you're not the customer you have no way of knowing what's important and what's not important about the product.** [anonymous HP client, quoted in "HP Corporate Quality 1989"]

From the Client's Point of View

A program is inseparable from the manual, sales brochure, packaging, delivery, training, support, and installation that come with it. In the same way, an application system is inseparable from the operating system that it runs on and the 4GL that it is coded in. All of these elements go into creating the client's experience. It takes a total team effort to ensure high quality.

For example, disposable contact lenses come in a plastic package that keeps them sterile and moist until use. However, peeling off the foil seal leaves a sharp edge that can cut your hand. The contact lens may give perfect sight, the marketing may be superb, the sales team helpful, and the product distribution speedy, but if the client cuts his hand opening the package, that undercuts the quality of the entire product.

It is frustrating to dedicate two years to a computer program, as happened to a former HP employee of my acquaintance, then find that the company bureaucracy cannot deliver the program to the users for another three years. The best people quit and form their own companies -- the ones who stay often shrug and say "that's not my job." At Robelle, we hold staff meetings every two weeks in which sales, support, finance, R&D, marketing, and administration provide each other with a global picture of product concerns and customer priorities.

Break Down Barriers to Clients

Programmers Should Take Technical Support Calls. There is nothing like hearing directly from an irate user of a piece of software that you wrote to motivate you to improve it.

No Matter How Much It Hurts!

Programmers Should Visit Client Sites. When you are on site, people mention problems that irritate them but which they won't call about. You see them use your product in unexpected ways and use ingenious workarounds for unsuspected design flaws. Users group meetings are another good place to meet clients.

Bug Tracking Software. There is nothing worse than ignoring a cry for help. Once you have a reputation as a "black hole" (bug reports are never heard of again), clients stop looking to you for solutions, or products. We use electronic mail and a keyworded database to collect trouble calls and route them to interested parties, including programmers. Anyone can append comments to the original call report and **the comments are distributed via E-Mail also.** This "conferencing" makes a lab programmer feel almost as if he is on the technical support line.

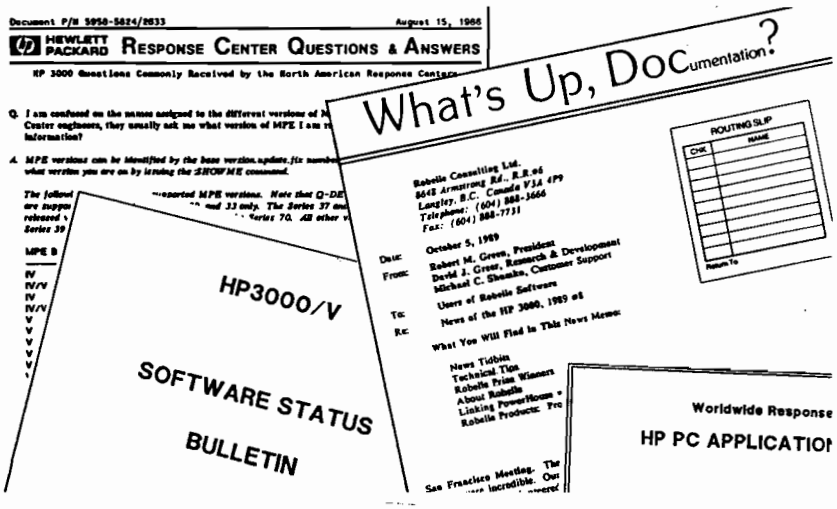
```
KBINQ  v 3.30 FRI, NOV 17, 1989,  2:49 PM

Ref:        288
Company:    RacRillan Cloedel Inc.
Caller:     Bobin Bennie
Phone:      (904) 661-8544
Serial#:    3100
Subject:    Batch/NS/Eating job lines
Product:    QEDIT 3.7
Status:     Open

New:  DAVID GREER        Batch/NS/Eating job lines  22 Mar89 10:36

* Orig:  DAVID GREER                                 22 Mar89 10:36
Bobin has installed Qedit 3.7 on two of their development machines. Qedit
is failing in many job streams, because it is trying to do teminal
recognition when run from a remote job stream. I'm sending Bobin Qedit
3.7.1 which we belive fixes the problem in most cases. Do we know whether
we correctly detect batch-mode when remotely run from a batch job in all
cases?
```

Continuous, Intelligent, Honest Communication. A dependable, concise technical newsletter builds client loyalty. It should be **selective.** Clients are busy and don't have time to read everything they receive every week. The HP Software Status Bulletin contains every known problem, but doesn't highlight the disastrous ones. This would require a senior person as editor. A newsletter should be **timely.** If you insist on a glossy four-color format that makes production stretch out to weeks, your news will be dated. We aim for one-day turnaround.



Admit That Mistakes Are Inevitable

> **The most important lesson to be learned from this incident is that even highly talented programmers make disastrous mistakes. Morris's program was comparatively small and simple, and had a limited and well-defined purpose. It faced no serious obstacles in the form of security barriers that attempted to foil it. Morris is known to be extremely skillful, was highly motivated to write an error-free program, and was not working under the pressure of any deadline. Despite all these facts, his program contained a catastrophic error [causing it to replicate and propagate itself far more rapidly than he apparently intended].** [Ornstein]

I found this example in the ACM's comprehensive review of Robert Morris's famous Internet Virus that invaded 6,000 Unix systems on November 2, 1988. Producing quality software is difficult and mistakes are inevitable -- not just mistakes in programming, but more disastrously, mistakes even in identifying what the client needs.

Be a Humble Manager

One of the key papers in the history of structured programming was *The Humble Programmer* by Dijsktra. Most programmers have accepted his approach:

> **I now suggest that we confine ourselves to the design and implementation of intellectually manageable programs... We shall do a much better programming job, provided that we approach the task with a full appreciation of its tremendous difficulty, provided that we stick to modest and elegant programming languages, provided that we respect the intrinsic limitations of the human mind and approach the task as Very Humble Programmers.** [Dijsktra]

Now we need to extend those insights to the equally difficult task of managing "what" programmers program. Humble management is like defensive driving. You must assume that something will go wrong at the worst possible moment and be prepared to switch gears quickly.

Some Techniques for Managing Error

Explicit Design Criteria. Programmers need guidelines to help them make difficult tradeoffs while programming. At Robelle, our criteria are reliability first, then compatibility, performance, and finally features. Without such leadership from management, programmers cannot be expected to produce a consistent and dependable style of program.

The Development Diary. This is a computer file that acts as a lab notebook; it records your thoughts and plans as you work on the code. We make entries for each day, with the most recent day at the start of the file. Other sections of the file list outstanding bugs, enhancement requests, patches for known problems, and documentation problems.

Batch Jobs to Test for Stupid Mistakes. If you touch the code, you may delete a line by mistake. We do automatic batch regression testing of each new version. The

tests are designed to abort if anything goes wrong, or update a results file if they make it to the end. We often run the test suite every night to check that day's changes. Ideally, each bug uncovered should be verified with a batch job that reproduces it. Once the bug is corrected, the test will pass and will ensure that old bugs do not creep back in by accident (this has happened to us!).

Frequent User Testing. The biggest danger is that you will deliver a working system that doesn't do what the client needs or wants. Even when your code does what they want, it never does it exactly right on the first pass. To minimize this risk, we send pre-releases of revised software to selected clients about once a month. A pre-release program is much like a regular release, including updated manuals and on-line help. We seek out clients who will treat this software as harshly as they would an official new product, and we often find them in our tech-support records. The benefit is mutual: we get objective feedback, and the clients often get very quick solutions to their problems.

Measure Quality in All Functions and at All Levels

What you measure is what you care about - people sense this. Measurements are good for recognizing achievements, not just detecting problems. Most jobs are repetitive -- it isn't easy to appreciate all the work you did in a year without numbers ("I filled 200 client orders", or "I wrote 2000 lines of code that got into production").

Start with concrete, mundane measurements that are easy to collect. They should be simple, they should relate to your long-range goals, and they should be understood by the people who make them. Just the fact that you are measuring will make a difference. If you show you care, people will refine the categories and numbers over time. Here are some you can start with:

Time to get a new release to the client, to fix a bug, to answer a fax.
Number of bugs reported each week.
Warranty costs, number of returned products.
Number of products shipped per day, percentage shipped in 8 hours.
Number of times competitor mentioned favorably!

Dan Warmenhoven has said that HP has two new metrics it will use to measure software quality: "... the number of post-release defects in the first year ... and the number of critical and serious open problem reports." [*Interrupt* Oct. 89].

Where Does the Money Flow in Your Organization?

Do salesmen get bonus trips to Hawaii and cars and fancy offices and awards and special dinners, while technical support people get a certificate of appreciation? Is it any wonder that the products don't work right? It isn't what you say that matters, but what you do. How is the budgeting arranged? Is there funding for on-going improvements of the software? Is your super-programmer allowed to work on a single project for years until it is a coveted tool of the clients, or is he constantly shifted to another project as soon as he delivers the first version, and replaced by a junior maintenance programmer?

Management communicates priorities every time it does anything -- whatever management spends time on and rewards is what the company will emphasize.

Concluding Remarks

The task of improving software quality is primarily a management task, not a technical one. The problem is not that we haven't adopted the latest, "perfect" system development methodology -- the problem is that we haven't been realistic about the immense difficulties in producing quality software, and we haven't been paying attention to the mundane, practical details of producing software that does what the client actually needs and wants.

I think that the primary responsibility of managers, the fundamental one around which all others will revolve, is to break projects into manageable steps, then deliver the new software produced by each step into the client's hands, so they can give you objective feedback for the next step.

References

Berger, S. et al, "Toward a New Industrial America", *Scientific American*, June 1989, Volume 260, Number 6., by Suzanne Berger, Michael Detouzous, Richard Lester, Robert Solow, and Lester Thurow.

Brooks, Frederick P., **The Mythical Man-Month**, Reading: Addison-Wesley, 1975.

Dijkstra, E., "The Humble Programmer", 1972 Turing Award Lecture at the ACM Annual Conference, Boston, on August 14, 1972.

Feynman, Richard P., "Personal Observations on the Reliability of the Shuttle," **What Do You Care What Other People Think?**, New York: W. W. Norton, 1988.

Halberstam, David, **The Reckoning**, New York: Avon, 1986.

Kohon, Michel, "Introduction to Step by Step", **SMUG II Proceedings**, Langley: Robelle, 1982.

Ornstein, Severo M., *Communications of the ACM*, June 1989, Volume 32, No. 6.

Moon, Richard, "Managing 4GL System Development in the 1990's", *Conference Proceedings of the HP Computer Users Association*, Brighton, England, July 1989.

Peters, Tom, **Thriving on Chaos**, New York: Harper and Row, 1987.

Rand, Ayn. **Capitalism: The Unknown Ideal**, New York: New American Library, 1966.

Schlender, Brenton. "How to Break the Software Logjam", *Fortune*, September 25, 1989.

How to Commit (or Prevent) Computer Crime

Louis R. Mills, CDP

Bio-Rad Laboratories, Inc.
1000 Alfred Nobel Drive
Hercules, CA 94547

(415) 724-5613

Do you like money? Would you like to make $50,000 tax free this year? Is a million dollars in a Swiss bank account worth a few hours work each week? Then perhaps you need to consider a career change to Computer Crime.

Consider this quote by Malcolm Senn, taken from a 1984 Omni Magazine article by Roger Rapoport:

> If I had been a traditional embezzler, they probably would have arrested me within six months. But with the computer, I never had to touch the cashbox. Being the Controller made it easy. First, I set up fourteen phony suppliers. Then I programmed the unit to pay automatically for nonexistent goods and services from my dummy companies. That way, I could be skiing in the Alps while my California employer mailed checks to these fronts.
>
> The system earned me a million dollars over six years. By then, I was ready to quit and start enjoying the houses, planes and boats I'd accumulated. I knew if I just left, my successor eventually would figure out what I'd done. Since I can't stand suspense, I began leaving clues to help the auditors catch me. I wanted my day in court, serve a brief sentence at a country-club prison, and live off of the money I'd stashed in Switzerland.
>
> But the auditors were impossible slow. Finally, in desperation, I started bouncing checks. It still took them another three months to catch up with me.

Malcolm was caught, tried and convicted. To his surprise, he spent five years at San Quentin prison before being paroled. He did not return the money. While in prison, he was the warden's assistant. He arranged the installation of a computer terminal so that inmates could take a computer programming class. Many of his students are now out, in society. Is one working in your firm?

Malcolm had no criminal record, had never stolen before, and was a trusted and valued employee. In fact, he fit the profile for computer crime perfectly.

The Federal Bureau of Investigation (FBI) uses a broad definition of computer crime:

Any crime where the computer is either:

A. The vehicle to commit a crime
B. The victim of a criminal act

Why choose a career in computer crime? Why not just rob banks for a living? Well, according to the FBI, the average bank robbery (1987):

1. Nets $6,325
2. Is quick. Under two minutes spent in the bank.
3. Dangerous! Armed guards and police.
4. Lots of physical evidence (Smile!) and witnesses.
5. 100% of these crimes are reported. (It's the law.)
6. 60% conviction rate, with few suspended sentences.

Now a computer criminal, on the other hand, has these FBI statistics going for him. The average computer fraud:

1. Nets $50,000+ (The FBI only handles $100,000 up.)
2. Takes 1 to 5 weeks.
3. Virtually no danger.
4. Little or no evidence -- you can erase it yourself.
5. 1% to 10% reported -- FBI estimates 2%.
6. 1% to 10% convicted, often for a lesser but included offense. Suspended sentences not uncommon.

So a little math tells you that if only 2% are reported, even if 10% of those are convicted, you only have one chance in 500 of either doing time or paying a fine. Not bad odds.

But good managers set high goals for themselves. Let's set some goals for you. Let's take a 1987 example from Volkswagen in Europe:

1. Net of approx. $259,000,000.
2. Took less than one year.
3. No one was physically harmed.
4. Little evidence was found.
5. One insider charged, seven outsiders indicted.
6. One conviction so far.

But some of you are real movers and shakers. $259 million is small potatoes. Let's set our sights on some real money. What has been the biggest scam so far? Equity Funding Insurance:

1. Fraud amount $2,100,000,000.

2. Built up over three years.

3. No danger.

4. Less than 10% of fraudulent entries found.

5. Many executives and workers charged with fraud.

6. 13 convictions of senior executives in federal court under the Foreign Corrupt Practices Act.

How do people get away with crimes like these? Won't other people turn you in? The reality is society's attitude about ethics. Ethics are often "situational". And if people don't perceive a direct harm, they often ignore criminal acts. It was not uncommon 10 years ago for graduate classes in Computer Science to give credit to students who could violate the security on the school mainframe. When a professor was asked about this at a National Computer Conference, his reply was:

"We're not here to teach ethics; they should have learned them before they arrived on campus."

Most of us in the computer industry often wonder about the ethics of vendors' marketing claims, pre-release announcements, the credibility of consultants and repairmen, etc. There has been a surge of interest in ethics in American business and MBA programs since the "insider trading" scandals of the last few years, so we may see a renewed interest in this area. The Summer 1989 issue of DPMA's Information Executive carried four articles on ethics.

But a typical computer crime is a conspiracy, and this case is a good example of two ethical failures:

In Chicago, a company president, the DP manager and a programmer were charged with a $40,000,000 fraud for inflating the inventory records to cover up poor management. The company board of directors decided to slowly deflate the figures to avoid a drop in the firm's stock price.

Both the Securities and Exchange Commission (SEC) and the Justice Department said "NO!"

The good news is that computer crime seldom closes down a business. Dun and Bradstreet provided these figures in 1987 for the reasons businesses fail:

0.3%	Fraud
0.6%	Neglect
1.3%	Disaster
9.8%	Lack of management experience
9.9%	Lack of experience in the line or industry
23.8%	Unbalanced experience
54.3%	Incompetence

Most of you know someone in upper management who fits into one of those last two categories.

But even more interesting to me are these figures from the US Justice Department:

80% of frauds start from unintentional errors.

15% start from "What if ..." scenarios.

5% are purposeful acts.

Computer Crime can be defined in three different categories and we will review two case studies in each to give you a feel for the areas you may want to pursue.

(Case study information is not published since it deals with techniques used to commit a criminal act. The publishing of such information is a crime in many places.)

1. Physical access crimes
   - A. Impersonation
   - B. One-dimensional security penetration
2. Transactional crimes
   - A. Financial fraud, embezzlement, etc.
   - B. Theft of goods or services
3. Electronic (programming) acts
   - A. "Salami swindles", etc.
   - B. Sysop or "techie" techniques

Computer crimes, like many White-Collar Crimes, have a low reporting rate. Remember the FBI estimate of two percent? Why?

1. Embarrassment felt by upper management.
2. The impact of loss of public trust in the company.
3. The potential effect on the company stock price.
4. Fraud suspected, but not provable or detected.
5. Culprits are unknown.
6. Legal difficulties from:
    A. Lack of evidence
    B. Conflicting laws or enforcement agencies
    C. Technological ignorance of management, police, judges, juries, etc.
    D. Jury sympathies
    E. Seizure of business records, disks, computers...

If you're in a company that is the victim of a computer crime, what is the recommended order of bringing charges?

1. Foreign Corrupt Practices Act
2. Federal/state fraud or embezzlement statutes.
3. Theft of goods or services.
4. Vandalism or sabotage.
5. Trespassing
6. Computer crime laws.

In the US, there is the Federal Computer Security Act of 1987 that makes you liable if you were involved in the design, implementation or use of a "dis-approved" computer application. So, if the system is certified for security, it isn't your fault if you broke into it, is it? Have them bring charges against the guy who designed it.

Don't fire or demote someone who is suspected of a criminal act. In many states, your action is illegal.

The following **management recommendations** are common policy in many organizations:

1. Keep DP staff off of production systems.
2. Identify employees with harm potential.
3. Restrict physical access to media & equipment.
4. Require pre-employment screening.
5. Vacations must be taken.
6. Rotate job assignments.
7. Use decoy names in address files.
8. Don't let people work alone.
9. Build in audit trails and controls.
10. Use exception reporting, highlight unusual activity.
11. Be ready to PROSECUTE!

Here are some specific **Security suggestions.**

1. Avoid high risk people in key jobs.
2. "Fair and Equal" treatment must be the perception.
3. Educate and drill on security.
4. Manage by walking around & talking.
5. Track severe personal problems without violating an employee's privacy.
6. Shred or burn your own trash. Don't let it sit out.
7. Require and verify proper ID.
8. Verify employment and education claims.
9. Know the local gendarmes and authorities.
10. Know what constitutes "best evidence".

There is a wealth of material about computer security and computer crime. I recommend any book by the following:

Donn B. Parker, Stanford Research Institute, CA

August Bequai, Attorney, Washington, DC

Jack Bologna, Computer Protection Systems, Plymouth, MI

Additionally, the following are excellent sources:

1. Computer Crime Legislative Resource Manual, US Dept. Justice, Bureau of Justice Statistics. 150 pages, price unknown. Read this first, before you take a suspect case to the authorities.

2. Guideline for Computer Security Certification and Accreditation. NBS FIPS-PUB-102, 95 pages. Nat'l Technical Info. Service, Springfield, VA 22161. $11.50

3. Computer Security Digest, CPS, Inc. 150 North Main St., Plymouth, MI 48170. $110 for 12 monthly issues.

For the current status of state or federal regulations:

Joseph Collins
DPMA Headquarters
505 Busse Highway
Park Ridge, IL 60068-3191

COMPUTER COMMUTING

Jean A. Hills, Vice-President
N. A. Hills Computing Services Limited
336 Piccadilly Street,
London, ON Canada N6A 1S7
(519) 672-1731

It is 8 a.m. and I am sitting behind the wheel, behind a car, behind a truck, in a traffic snarl-up which stretches miles ahead and trails miles behind. There is no way of escape because there are cars to the left and right of me. The honking has ceased and I have seen the police speed up the median followed by an ambulance. Someone up front has taken the time to stage an an accident to the accompaniment of honking horns and with a captive audience of stopped cars.

Certainly I feel sad to think that someone is hurt but I am also frustrated by the fact that I have wasted 30 minutes already in this line-up and will probably waste another thirty before the mess is cleared away and traffic returns to normal.

It is a warm day and I open the window for some relief. I choke on the exhaust fumes generated by hundreds of vehicles. I close the window and turn on the airconditioner. Even it seems to suck in the fumes. My throat is burning. My eyes smart. My head feels like it is being hit by a sledge hammer. When will the traffic begin to move.

This is not a rare phenominon. It is something which happens at least once a week. Even at the best of times commuting takes at least twenty minutes of my day. I settle down to wonder, wait, and waste time. The automobile that promised us individual freedom to come and go as we pleased, has become a commuter's prison.

Just imagine that at one and one-half hours per day used for commuting, calculate how much of your life is spent this way. 1.5/15 is about 10% of your waking hours you spend this way. If you work for 45 years this means approximately 4.5 years will be spent commuting. Is this the way you want to spend your time?

One of the main functions of each new machine is to save time, to save money and to be more efficient. Each new invention promised us greater freedom from monotony and drudgery. It also promised more for our money. Household appliances have freed us from washing clothes, washing dishes, beating rugs, chopping onions and so on. Today one can manage a large household with a minimum of human assistance. When I queried the original owner of our home, he said that there were four permanent staff and four occasional workers. This is quite a contrast to my half day a week cleaning woman and my electrical appliances.

Both on the farms, in the home and in the factory, machines have improved the quality of life. Today we face a new challenge that the computer offers us. If we can leave our traditional ideas behind, we can use this machine to eliminate the large volume of traffic that each of us faces every day. The computer can do the commuting for us.

With a telephone line and a terminal, my work could be done at home. The hours I spend behind the wheel could be productive hours at the keyboard. The waste of life and suffering caused by accidents could be avoided by staying at home and working on a computer. Exhaust fumes could be lessened by fewer people commuting to an office. The terrible urban sprawl - the tremendous waste of valuable farm land for offices and skyscrapers would be halted. Let us consider some of these ideas separately and then let us consider how the computer would contribute to the solution of some of the world problems.

Two of the most pressing world problems today are the ecology and the family. Both are changing at an alarming rate and the changes are not the best. It seems to me ridiculous for us to persevere in an archaic lifestyle when we have the means to change it for one better. Change is a must because without it the human race and our planet will not survive.

Throughout history there have been major inventions which have necessitated a change in lifestyle to perpetuate the human race. The first involved cultivation of the land, agriculture. People no longer continued to live nomadically, they could stay in communities and share the responsibilities of their primitive life. The next was the printing press which began the age of literacy and people could exchange ideas through time and space on the printed page. The next was the invention of machines and the use of power to propel these machines. The industrial age produced more goods with less manpower. Today we stand on the threshold of the computer age which offers even greater profit in time and money. It offers us release from the drudgery of mundane, monotonous tasks. The computer can not only do repetitious intelligent thinking operations, it can propel robots to do menial work as well. Your imagination is the only limit to what a computer can do. For it takes imagination to see the possibilities and then it takes expertise to fulfill these dreams.

One of life's most fulfilling jobs is that of raising a family. In today's economy it is almost impossible to raise a family on a single income. Both partners must work to contribute to the upkeep of home and family. But what is happening to the children if both parents are away from home working? As teachers, we know the latchkey child spends much time alone and before that as a preschool infant, the child is hustled off to a nursery school to while away the hours until one parent returns.

As one daycare person expressed it, "We see the first smile, the first step and we hear the first word." What a price to pay for the young parent. However there are ways around it if we just look.

Why do we not assign work to be done in the home? A child's needs can best be met by a full time parent and why not have the office work done at times that will not conflict with child care, performed in the home, using a terminal with a data line to the main processor.

Why should a woman be denied the right to pursue her career? She should not. But let us not bury our heads in the traditional ideas of an old-fashioned concept of employee supervision, that is, that work must be done on the premises, between the hours of 9 and 5. Why not do the work from a workstation at home between the hours of - well 1 and 24, whenever the time is available. Why not allow at least one parent to be home to comfort and share time with their child during these critical years. Work could be done during nap times, or in the evening. School age children need a parent home before and after school, and to supervise the lunchtime. During this period work will be done in three hour shifts. Why does it matter when the work is actually done as long as it is done efficiently and on time. Why not let the machine fulfill its purpose of time saving, and efficiency, and allow the family to grow and flourish.

A child needs a consistent, caring and accessible parent during the early years. A teen-ager's needs vary but there must be parent accessiblility at this time. I remember a young teenager in my class, who was forever phoning one parent. The one was a surgeon and most often the child had to settle for the secretary. The mother was a reporter and was often away on assignment and even though long distance calls were accepted, she often could not be reached. Here the housekeeper had to be the substitute. Substitutes are good but they do not have the same vested interests as the parent. They are fragmented and less effective because they do not know or understand the totality and the enormity of the child's needs. Even a paid professional does not have the personal commitment demanded by the child. I remember watching a group of nannies in New York City's Central park. They pushed the prams and walked the little ones but they were more involved in their own social chatter than the activities of their charges.

The reasons for change are evident all around us. Our trees and plants are withering and dying from exhaust and acid rain. Our children are being shunted from one agency to another and cost of child care and tremendous staff needed to educate children of this nuclear age is escalating each year. One little boy, from a highly motivated professional family might need one or more of these; a psychologist, a psychiatrist, a special teacher, often an aid, a resource teacher, the police force, family counselling, the family

and childrens's services facilities, and later will involve hospital and clinical services. This is just one example of one child's need and most of the cost is borne by society.

In London, there is a movement to move business into the home. Young professionals are building homes which will accommodate an office with a computer so that one or even both parents could work most of the time in the home and thus provide a parent supervised home life for their children.

As a teacher I have always been aware of the difference in development, in achievement, in attitude, in behaviour and in social skills, of those fortunate few students who go home to a parent figure in contrast to those who carry a latch-key.

Why not provide the secretary, the architect, the designer, the researcher, the writer, the office worker, the programmer, in fact any profession that requires record keeping, repetition, research, and accounting. This could eliminate large government offices involving taxation and statistics.

Fifty years ago people relied on the Sears cataolgue and in Canada it was the Eaton's or Simpson's Catalogue for shopping. Why not computer shopping today? The list of possible innovations the computer is capable of, is as long as your imagination will stretch. It will require divergent thinking and sometimes untraditional activities to promote such a plan.

If the time we are wasting in traffic lineups and in supervising people at work, was tallied, it would be astromomical. Just think of the amount of creative time and money we are wasting with our concept of commuting to the office. Most of the manager's supervision time could be used for creative work and the production of stimulating challenges for the employees. Removing the supervision barrier would allow a better exchange of ideas and more cooperation within the team. The introduction of the computer commuter would be a saving to both employee and manager.

If we are to move into the twenty-first century with improved social patterns, brought about by a renewed family management system, we must utilize the computer and its ability to communicate interactively across time and space to allow not just quality time for our families but appropriate time.

In my classroom we have three terminals, two are P.C.'s and one is an H.P. connected to a mainframe at our home. When my husband wants to send me a message or ask me a question, the terminal sends a beep and the message appears. At first the children thought it was a message from outer space but

gradually they have become used to this procedure for communication. One nine year old boy was so intrigued by the process that he combed the manuals to discover how it was done. One morning, on 60 terminals across southern Ontario, Ben sent a cheery greeting. At first the staff thought they had a hacker in the system but Ben honestly signed his message so it was no problem to trace. We were lucky he did not interrupt a long printout or interfere with any serious work. Ben was firmly reprimanded on the seriousness of his caper and life in the classroom returned to normal.

If a nine year old can send messages across the province, if I can communicate with my home from school, then isn't it possible that an employee can work at home and using a computer communicate with the home office, with buyers and sellers(telemarketing can become Compumarketing),could design and create great buildings or bridges, could compute in the millions, and be able to stay out of the traffic jams by working at home and be available for family consultation. Stretch your imagination beyond the obvious 9 to 5 office hours plus at least 2 hours of commuting time and see what fabulous ideas you can come up with.

Today, I can sit at a terminal in Las Vegas and talk to my office staff or my students in London, Canada. Tomorrow I will work effectively in my home office and contact my clients across the country.

How would this type of operation be implimented? It could be a network, or a telephone line, or a flopy disc sent by courier to its destination. In our office we use a telephone line. When someone on staff has a family situation, deliveries to wait for, a sick child, we let them work at home. In this way the work can be continued and uninterrupted and at the same time the home situation can be handled. In fact our office manager has a permanent line and terminal at her apartment where she often works so that she can work uninterrupted by staff queries and telephone calls. She gets much more work done and she can work in her own time without losing time for travel and interruptions.

If we eliminate travelling to and from the office, we will reduce the number of traffic jams and accidents. We will reduce the amount of carbon dioxide in the environment. We will reduce the wear and tear on our vehicles and thus reduce the number of vehicles needed each year. This in turn will reduce the output of acid rain on the environment and will promote a cleaner ecology.

If we work at home then we will eliminate the need for more tall office towers. We will eliminate the large parking garages and tarmacked lots. With need of fewer offices we will have more parks rather than tall buildings. Trees and foliage provide more oxygen and enhance our ecology system.

How do we monitor our work? If the only function of the manager is to monitor arrival and departure times, then this is a misuse of managerial time and training. People work better without the pressures of big brother watching them. The true role of the manager is to see that work gets done, to promote confidence among the staff and to set goals and objectives for the team.

It will be important for this new manager to be able to set realistic goals and guidelines for each job to be done. Then it is the responsiblity of the employee to meet these goals and objectives. In this way a subordinate will be evaluated by productivity rather than appearance. A true professional does not need to be monitored. Most people take great pride in doing their best.

In my classroom, there are three computing terminals. Each is equipped with a personal record of the work done by each student. In this way I know how long and how much work the student has done, what calibre of work is done and I can trace the culprits who break into other students programs. If this can be done in school, why not use the same technique for business.

Working in your own environment is also condusive to work enjoyment. It reduces the amount of time wasted at the coffee machine, it eliminates the office gossip grapevine and it gives the worker freedom from the pressures of supervision.

Of course there will be people who abuse the system. Even in the office, there are the daydreamers who simulate work, the office chatterings, the long periods away from the workstation for coffee breaks. I was amazed to be in the Bell Telephone office waiting to pay my account at opening time. I was almost run down by the crowd rushing to the cafeteria for their morning coffee and they had just arrived at work.

Working in your own workspace, in your own environment, at your own time and pace induces greater pride in work. The one thing I enjoy most is being away from my office classroom. Taking the children outside to a park, to a museum or some other place to work is much preferable to always working at the workstation. We work longer hours without supervision than when I have a supervisor who is checking to see what we are doing.and when we are doing it. Field trips take extra time and preparation but nobody complains and my principal has to accept the fact that even if he can not see me, we are working.

With the proliferation of large office buildings, with centeral air conditioning, one of the greatest health hazards is the stale, polluted, air which is circulated through the offices. Most windows are sealed so that no fresh air can get in and many air-conditioning units just

recycle stale air. Woman's Day, a popular woman's magazine reported that this stale air is one of the greatest health hazards. We have all heard of Legionnaire's syndrome, and it is present in many large office units.

How can this working at home be effective? It will be important to point out the advantages to the workforce. A saving in commuting expenses, a saving on clothes, healthier working conditions, and more than quality time for the family. Freedom from office pressures and a resultant contribution to saving the environment with fewer cars on the road.

With these benefits and better working conditions the responsibility for making it work will be with the employee. The honest ones will benefit by this arrangement and the irresponsible ones will soon be without a job. They will no longer be able to vegetate in front of the terminal, or take root at the coffee machine, they will have to produce results that will satisfy the manager.

There will always be a need to meet with the staff at some central location. Architects have to discuss with their clients, lawyers will have to appear at court, writers will have to appear at social functions, teachers will have to have some personal contact with each student, accountants will have to consult with their clients and so on, but there is the the possibility that the computer will create a whole new work environment and work ethic. With the help of creative people we could limit the destruction of the environment, build fewer office towers and keep traffic to a minimum. Instead of sweating it out behind the wheel each morning it would be possible to to get several hours of work done before breakfast and then have the time to enjoy the kids before they head off to school.

The change of venue from office to home can result in a healthier work climate, minimize stress, eliminate office gossip, and enrich the lifestyle of the family. In addition, it will reduce the need to commute by automobile and thus create a healthier ecology. It will relegate the car to it's original purpose - providing recreation rather than becoming a commuter's prison. There is so much to gain, I find it hard to understand why more people aren't thinking, talking and promoting the computer commuter.

ENVIRONMENTS OF SUCCESS

George B. Scott
Great Business Solutions, Inc.
P.O. Box 950
Graham, WA 98338-0950
(206) 847-8924

Over the twenty-five years that I have been involved in business, several experiences stand out that were quite successful. Each of these experiences occurred within an environment that had many striking resemblances to the others. Analyzing the various situations of success has led me to believe that certain criteria will not only produce success, but will deliver success spontaneously. This paper presents four situations which have led me to these conclusions. I hope you enjoy this trip into the fairyland of great ideas which seldom find a home.

The first situation was one in which I was a young chemist working in a research and development laboratory. During this tenure, I was given the opportunity to lead the development of new products for an existing product market. In addition to this responsibility, support of current products and manufacturing operations related to current products was expected to proceed as normal. At the time, the staffing of this lab consisted of myself and two additional highly qualified technical gurus. During the three years that I headed this operation, twenty-seven (27) new products were introduced and our sales expanded from 9 to 17 million dollars. One might say we were just lucky; however, I believe several environmental conditions existed which greatly augmented this success. First, almost no direction was ever given by any manager as to how we were to solve any technical problem. Essentially, we were give carte blanch with regard to developing new polymers, new production procedures, new fields of chemical types to be used in the particular application area of interest to our company. Secondly, we had a significant amount of freedom to spend our time as we saw fit. All we had to do was produce results, which essentially delivering new products which the competition didn't have and increasing our sales and profits. I should note that we increased our profit margin for the product area to approximately 100%, which meant that our contribution to the bottom line exceeded that of all other lines whose sales exceeded 150 million dollars.

The second experience in which I was involved that had remarkable success was also involved in the chemical world. Don't worry folks, this will all be related to the how you can use this information in your own jobs before we are through. In the second situation, I was involved with an R&D department for a major chemical company. At the time, we were having some problems when products were transferred from the labs into production. To help alleviate this situation, I was given the opportunity to establish

a team which was to make sure that new products and processes developed in the laboratory environment could be transferred into production without causing equipment failures, severe production problems or even injury. One of the advantages of this opportunity was that I was authorized to select from the R&D environment the cast of players to accomplish this. Since I had been working with all the candidates for a couple of years, the team was built with diversified technical talents and expertise, but with exceptional compatibility of personalities.

Shortly after starting this endeavour, which included building a pilot plant and installing equipment which we selected, it became apparent that we could achieve our charter and still have significant time left which was not allocated. As a group, we discussed what we might be able to achieve with all this equipment, technical expertise and spare time. We decided to study the entire production situation and determine where we could make the greatest contribution. First, we determined which products accounted for the greatest amount of production time per year, using the theory that a small improvement in a large volume product would provide more to the bottom line than a large improvement to a one time a year product.

The result of this effort was that we reduced production costs for standard products by over 2 million dollars a year. In fact, the company was able to expand production for expanded demand and new products without having to buy land and build a new plant as had been planned. In short, we were heros.

In reflecting how this came about, it is very obvious to me that one of the environmental factors was almost total freedom as to how we were to achieve our charter. The second factor was the freedom we had as a group of talented technical people to determine how best we could use our expertise to help the company and do something which excited us.

The third experience in which I have been involved which has had exceptional success was one that the Chairman of the Board of the corporation for which I was working recommended that I consider. This situation was one in which an organization call the "Resource Center for the Handicapped" was being put together. One of the objectives of this organization was to select and train people with severe physical handicaps. Quadriplegics, paraplegics and blind people have been among the students trained by this organization.

During the establishment of this organization, CEO's and Presidents from many of the corporations in the Pacific Northwest selected a member of their corporation to meet together and form a Business Advisory Council (BAC) to help make this idea into a reality. I remember very well our first meeting in which approximately 50 highly talented people from a variety of businesses met to

determine what had to be done and how we could transform this idea into a reality. Think of this. Approximately fifty people who don't know each other from fifty different companies meet to establish an organization which was to start training students within a few months. Within a very short time we had organized into a group of committees such as student selection, equipment, curriculum, evaluation and placement. Each of the committees then met and determined how to best meet the responsibilities as divided among the committees. Each of the committees selected a chairman to coordinate with the other committees. The entire BAC met regularly to discuss how best the organization could solve problems and achieve goals associated with the organization.

Today, the Resource Center for the Handicapped in Seattle is considered the model organization in the nation and is being used on an international basis as a model to help return severely handicapped people to productive lives in the mainstream of society. Again, let me note that one of the dramatic environmental conditions faced by this group was the total freedom as to how we were to achieve our goal. We were allowed to establish our own organizational structure and divide ourselves so that we could participate in areas which were of personal and professional interest to us as individuals. As a result, friendships were established which have lasted through the years and the support to the resource center from the business community has grown steadily.

Finally, as we look at the world situation today, we see one environment in which the people have been entrusted with freedom to decide how they want to contribute to their society. Another environment had a few top level officials decide what people were to do. We have seen the results of how a free group of people can produce using their ideas and freedom to the collective good at a rate that overwhelms that of Russia and her satellites. Again, we can see that the ingredient of freedom when mixed with talent can achieve great progress; whereas, severely controlled environments rarely encourage any of the participants who are being controlled to do anything more than the minimum required.

As a result of these experiences, I have selected the following list of factors for success:

1. Successful organizations are those who best enable their participants to use their capabilities and creativity.

2. Participants are more creative and productive when entrusted with freedom.

3. Desire and interest are stronger motivator than assignment.

4. The need for technology transfer is so great that it can only be accomplished by self-motivated "champions". Champions can only survive where there is support and freedom. To unleash the champions, freedom must be substituted for control.

5. Management must encourage champions to come forward i.e., induce the genie to come out of the lamp.

6. Successful teams are those in which the champions establish team members who are mutually attracted to each other. The "chemistry" is more important than the particular skill set.

7. People must be valued for all the skills and knowledge they possess as well as the potential yet undeveloped. Management must stop thinking of people as being only competent in the area of their current assignment.

8. Successful organizations are those who react quickly to changing conditions. Excited and enthusiastic team members who are entrusted with freedom will use their intellect and experience to deliver dramatic solutions much quicker than those who are "told" what to do and how to do it.

DON'T BE CRUEL TO A HEART THAT'S TRUE

Diane Amos, C.P.C.
Amos & Associates, Inc.
633-B Chapel Hill Road
Burlington, N.C. 27215
(919)-222-0231

As a manager, do you have employees whose "hearts are true"? How do you show them that you appreciate them? What do you do to retain those valuable employees?

In today's HEWLETT PACKARD marketplace, one of the costliest factors that a company has to deal with is staff turnover. The dollars start adding up when you look at the cost of recruiting a new employee, training them and then assessing lost productivity. The days of remaining in a job for twenty years to receive "the gold watch" are no longer reality, and with our mobile society being what it is, the 90's may show this to be an even larger problem. The average tenure of DP professionals is now 2 1/2 years. How can you, the manager combat this epidemic? When your good employee is resigning, should you offer him a "counter offer" to stay? NO! This is usually a drastic mistake for everyone concerned. By putting yourself in this position you are giving your employee the message that he's only valuable when threatening to leave. Let's take a look instead at preventive measures that will keep that employee from leaving in the first place. The best way to do this is to examine the numerous reasons that employees give for leaving companies.

The #1 reason for leaving a job is boredom and lack of challenge. Do you have your prize employee stuck in a corner because he does his job so well that you don't want to rock the boat? Well, the boat may sink with that attitude. People need to grow and expand in their jobs to be truly happy, and keeping a valuable employee in a "rut" will surely cause him to look elsewhere for challenges. It takes some insight and planning to keep your employees challenged, but the investment is well worth the time and effort. Think of special projects that you can assign, new and innovative ways to improve your shop. Cross training of jobs and rotation of duties keeps things interesting. In addition, many successful managers tell me that the more they challenge their employees, the more they are challenged themselves by their employees. Are you afraid to allow your employees to stretch themselves thinking that they'll outpace you? What may really happen is that you'll end up stretching yourself!

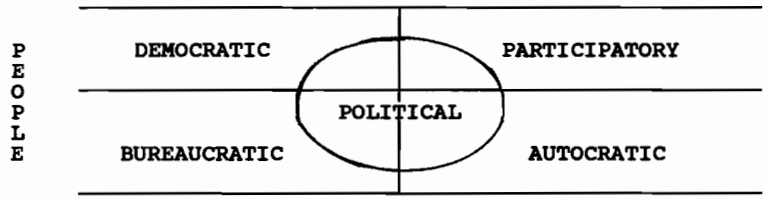
The #2 reason is lack of appreciation. When was the last time you really let your employee know how much his work, his effort and his attitude was appreciated? Do you wait for the yearly review or do you constantly reaffirm him with large doses of sincere praise when he's handed you a job well done. Management studies have shown that daily support, encouragement and praise for even the little things keeps an employee on an even keel. It's often the little things that count, and there are several small things that you can do for employees that may make a difference. Some ideas may be to take your employee out to lunch, or give him the day off. If your city has a baseball team, you might give him baseball tickets in his next paycheck. It's not the dollar amount but the thought that counts here.

Lack of recognition is a very similar reason. When an employee goes the extra mile to finish a difficult project or solves a problem that no one else has been able to solve, he looks for some <u>recognition</u> that his effort was noticed and appreciated. Again, you do not have to go to great expense to let an employee know he is valued in the company, as little things that are said or done often mean more than you think. A special parking spot for the "employee of the month" costs you nothing, but makes that employee stand out. Also, choosing the employee who is doing the steady, productive job for you to go to that next LUG, RUG or INTEREX meeting rather than the same ones who are usually chosen, may give that employee the feeling of importance that is needed. Recognizing an employee in a staff meeting of his peers with a plaque for "special service" costs next to nothing. A word of caution here: it is often common practice to "take turns" in recognizing employees so that everyone gets covered. This is a mistake! It diminishes the impact and the value of the reward. Recognition is meaningless when it's given for mediocre performance and when it becomes commonplace. A very productive way to recognize an exceptional employee is to let them prepare a presentation for a user group meeting. Start with the presentation within the department. This action will recognize the employee and also train the other staff members thereby killing two birds with one stone. A success story that I can relate is one of a rather quiet, introverted System Manager who once he warmed up to you was a brilliant, funny and witty fellow. His manager put him in the position of heading up a panel at a User Group. What a pleasure it was to see this employee blossom under this new position and swell with a feeling of importance. Being creative with ways to "recognize" your valuable employees <u>now</u> may save you the need of looking for creative ways to replace them later.

Another reason is a lack of input in the decision making process. If Joe has a job to do but no input on the best way to do it, his frustration level will grow. People want to participate and have a voice in the decisions that affect them. Of course, that isn't always possible, but spreading the decision making power to the lower levels will instill a sense of ownership, responsibility and accountability. As Nancy Austin, co-author of "A Passion for Excellence" discussed at the Interex Management Symposium: the companies that will grow and thrive in the 90's are those that will share the decision making power with their employees. "Trust" is a key issue here. Trust your employee to handle privileged information. You can't ask people to solve big problems without letting them solve little ones. Ms. Austin indicated that the companies who will have the edge will be those who nurture individual contribution, creativity and quick decision making. Critique your own management philosophy. Are you doing this?

Lack of direction from management is often heard as a reason to leave. While you're looking at your management style, check to see if you are being clear with your expectations and consistent in your demands. Monitoring your employee's progress on a monthly basis rather than at yearly review time will help to keep this in check and avoid room for misunderstandings. Have a two-way discussion to determine that you both are on the same wave length in how and when you intend to achieve your goals. Communication is the key, and having predetermined goals and objectives that you both agree on will avoid this problem. A performance plan should be in place and it's extremely important that the performance plan is a formally written document detailing the performance standards that the employee is expected to meet in order to achieve his goal. This then becomes the basis for performance appraisals which you can measure against. Employees love them, because if their boss dislikes them, they have something to put their hands around. It forces the boss to treat all his employees equitably and measure them by their performance rather than by emotion.

Matching your employees working style to your management style may help to avoid problems in the future. First of all, determine your management style. Managers should be both people oriented and production oriented. Look at the diagram:



Is your style Democratic where you have more concern for your people than for production? Is it Autocratic where you care more for production than for your people? Or is it Participatory where you're concerned for both? Or Bureaucratic where you don't care about either your people or production? Or is it Political where you flip-flop between all four styles at any given time? Having your applicant describe his ideal boss may help you to make that right hire and match the style at the very beginning. This may save you endless struggles of philosophical differences that no one can solve.

No career path? I hear this reason for leaving quite often! This is a difficult area in the HP market because there are so many small shops with few vertical career paths to offer. People tend to think of career paths as straight up when sometimes they're sideways and at angles. Discuss openly and freely with your employee what he truly wishes to do with his life and career. Perhaps you'll find that he could progress into the user area and would be happy to do so. Then put a career plan in place where he can take concrete steps to prepare for this next step. Don't make empty promises, but give a clear direction on how he can reach that goal. Make it a formally written plan detailing what your employee wants to do, and what he must learn or achieve to get there. Does he need to take a certain course or learn a particular skill? Where will he be able to go if he masters that? Be very specific and clear. What if your valuable employee wants to progress, but stay in a technical field and not go into management? A long term employee brings added value to the company and he shouldn't have to change jobs to have a career path. Raising the upper limits on salary and increasing the responsibilities and decision making input for people in the technical field could keep these employees satisfied.

2008-4

Don't Be Cruel To A Heart That's True

A poor working environment is often given as a reason to leave. As manager, take a hard, critical look at the workplace that you have to offer. The best way to do this is to actually work in it for awhile. Is it conducive to good work, and a place where people will enjoy coming to work, or does it leave something to be desired? I've heard horror stories of computer rooms with desks crammed into them making the employees feel like they were in jail cells; dark and dingy cubicles stuck off in a corner somewhere far from where they needed to be to be effective with users; and I've seen offices that should house 1 programmer have 4 people stuffed into them. Issues that should be addressed and looked at carefully are the temperature-(too hot or too cold?), noise level- (too noisy?), color-(drab and lifeless, or offensively unnerving?), smoke-filled?, music-(irritating or conducive to efficiency?), traffic-(people tripping over each other?) and the general esthetic nature of the place. These are often more subtle reasons for leaving but they work on the frustration level of an employee. Have you ever been bitten by an elephant? No, it's the gnats that get you, not the elephants.

Are you giving your employees valuable training and sending them to the necessary schools, or are you expecting them to pick up everything on their own? It's been said that people are most happy doing what they know how to do well. If they feel like it's a constant struggle to keep up because of lack of training, this frustration may turn into a reason to leave. Giving training is a good investment as it allows your employee to be more productive quicker than if he had to learn something from scratch. An important side benefit to his productivity is his feeling of putting to use what he learned and satisfying his thirst for knowledge. An inexpensive and productive way to do this is to provide in-house training by one of your own "experts". This idea cuts costs and recognizes a star employee all at the same time.

With all of the latest technology bombarding the HP market, a common complaint by employees is not being trained on the "latest, greatest". Employees want to stay current in their knowledge of technology and resent working on out-of-date equipment or unsatisfactory software. They know it's career suicide to do otherwise. Employees will jump ship just for the opportunity to work in a "state of the art" shop. As managers, staying current yourself with what's new and desirable on the technology front will keep you ahead of the game, make your shop an attractive place to come to and one that people will not want to leave.

"I'm stuck with a boss that hates me". What do you do when you have a personality conflict? IBM Corporation gives "skip-level interviews". This is where the boss's boss meets with the employees to hear what's going on. One benefit of this is for the senior manager to "hear from the horse's mouth" what's going on in the department, rather than only what the department manager wants to report. A critical benefit for the employee is that he has "an ear" for his complaints and an outlet for his frustration. Second-level management reviews of performance appraisals work well too. This is where the boss's boss sees the appraisal before the employee does, and then again after the employee is given the appraisal, but before it goes to Personnel. This offers good objective safeguards.

What if you have a "bad apple" who's driving everyone away? We've all seen this happen, and it always seems to take numerous sacrifices of excellent employees before management wakes up and does something about the "bad apple". Bad employees add to the work load of the good employees, and the good ones end up being "punished" for someone else's failures. This is sad to see, and it's often symptomatic of the manager's inability to deal with problems. As a manager, you can prevent this from happening by being decisive and cutting the rope when necessary. Keeping someone on board who spoils the rest of your staff is only going to cause you more problems in the long run. No one is indispensable and even if this "bad apple" is the most knowledgeable person you have on staff, cut your losses and do it quick! Remember, it's not the people you fire that give you trouble, it's the people that you don't fire.

Is your shop an unhappy place to work? Make it fun! Work does not have to be serious, and laughter cures all ills. I've seen more contented employees in companies just because it was a fun place to be, and everyone had a good time working there. Be creative! Start your next meeting with a video of "baseball bloopers" to get everyone in a good mood. And who says that a department meeting has to be in a conference room? Take everyone out to the park and have them bring their lunch. If you're in a coat and tie environment, have "casual day" every other Friday. On your way to work, stop by the 7-11 and pick up the big 50 cent bubble gum. That'll turn everyone into kids when each person compares how big their bubble is. Dress up for Halloween. Have a team golf or volleyball game. An owner of a company once said to me, "I feel that I'm in the entertainment business-making my people happy." As manager of your department, you can make this happen too.

2008-6

Don't Be Cruel To A Heart That's True

$$$$...Salary is a common reason given for leaving although I believe that it is generally secondary to the other reasons. Sometimes you just have to dig deeper to uncover the real reasons. Too often they only come to light at the exit interview when it's too late to remedy the situation. If someone is unhappy with his job, he may say to himself.."I'm not paid enough to take this abuse". But I've seen too many times where an employee will stay with a company for a long time, underpaid, simply because he is truly happy with his working conditions, treatment and job functions. However, let's address the salary issue. Is pay based on performance more than on seniority? It better be or you'll end up keeping your oldest employees instead of your best ones. Have you been staying current with the going salaries in the marketplace or have you fallen behind? Has the employee's job grown to where it needs to be reevaluated and a new worth established? Assess each job objectively and decide what it is worth. Then pay accordingly! An added possibility is the incentive bonus. I have a client who uses the incentive bonuses very successfully and reports a dramatic increase in productivity. He and the employee set concrete and clear goals at the beginning of the year, with several goals (8-10) to reach and then monitors the progress monthly. He says the employee has a sense of pride, ownership and reward when the goal is achieved, and the money turns out to be only a side benefit.

Let me leave you with a story. There was this data processing company that was seeking a technological breakthrough which would put their company out in front in their industry. Because this was a make or break situation for this company, the entire staff was hard at work to achieve this goal. One day the President of the company was in the lunchroom having lunch, when one of his technological genius's suddenly found the answer to the problem. He had broken through the barrier and discovered what they were all looking for! The President was obviously thrilled and excited beyond belief, and wanted to instantly reward this employee for this achievement. In his excitement, the only thing he could think of saying was, " Here. Take my banana!" Well, the employee was so pleased with being recognized by the President, that he put the banana on his shelf for all to see. It turned brown, started to smell, but he wouldn't throw it away. When the President saw what the employee had done with the banana, the symbolism hit him and he immediately went out and bought little banana pins. He went back to that employee and pinned the little banana pin on him. From that point forward, the 1 cent "Top Banana" pin became the badge of highest honor in that company.

The moral of the story is corny but real...don't be cruel to a "heart that's true" if you want to keep your "top bananas"!

2008-7

Don't Be Cruel To A Heart That's True

CASE, Design and OOP - A "State of the Art" Assessment

Robert R. Mattson
9545 Delphi Road S.W.
Olympia, WA 98502
(206) 736-2831
(206) 352-5038

Abstract

Each day brings new hardware and software "answers" to solve all our problems! This talk/paper focuses on the state of the art of analysis, design and programming in relation to CASE and OOP (Object-Oriented Programming). It will look at the fundamental problems we face in developing and maintaining quality systems and what role these two "answers" might play. The question that will be addressed is... whether these two solutions should be considered magic, myths or maybes?

Preface

CASE and OOP...what do these two buzz words mean? How much attention should the system professional be giving these today? Are they significant issues that the, already overloaded information system professional should be addressing? How do they relate to the real problems we face with achieving quality systems. How does one cut through all the hype? These were a few of the questions I needed to answer. So I set out recently to find some answers. I have discovered some very interesting information and even a few "answers!" It is this information and these insights that I hope will be of help for others with similar questions.

Our Challenges and "Answers"

There is a an underlying fundamental challenge here. The information systems business is over whelming these days. This feeling is "in general" and "in particular." In general this problem translates into a blur of all the new technologies, techniques and challenges which seem to be increasing exponentially while our human absorbing/learning capacity is increasing fractionally at best. This means we must be selective. It is impossible to focus on it all!

In particular, the last ten years has seen a multitude of "answers" to the challenges we face. A little brainstorming could come up with a list a mile long. Here are just a few.

Structured Analysis	RISC
Unix	Structured Design
Structured Programming	Data Modeling
Work Stations	C, Ada, Prolog
Transact, Smalltalk, etc	SQL
Expert Systems	Relational Databases
Fourth Generation Language	Information Modeling
End User Computing	PC's
Spreadsheets	Desktop Publishing
SAA	CASE
And More and More	

How many more can you come up with to add to this list? It seems that we are bombarded by easy "answers." We can add another one to the list. This is OOP - Object-Oriented Programming and its associated object-oriented analysis and design.

There are entire books, multi-day conferences and classes on CASE and OOP. It may seem a little audacious to attempt it, but my goal is to condense this mass of current information, outline the critical issues and point to their significance.

Before directly addressing CASE and OOP let's explore the general area of system design. We need to do this because many of the "state of the art" issues and conclusions about system design directly affect any evaluation of CASE and OOP.

System Design

What is system design? How do we recognize an excellent design as opposed to a mediocre one? What is the best way to assure an excellent design today? Those are very hard questions on which to put concrete answers. Let's explore some key ideas.

The purpose of doing analysis-design and programming is a successful system!

This seems obvious doesn't it? Surely, this most fundamental axiom is agreed to by all professionals. Assuming this, it is important to state it at this point because much of what will be presented later needs this idea squarely visible.

In fact, this "obvious" point needs to be emphasized here because it would seem that it get's lost many times in all the hype related to CASE and OOP. The tools and techniques seem to take on a life and purpose of their own. We need to have our goal clearly in mind in order to interpret the advertising from the proclaimed "gurus" and the vendors of method and products.

A successful system is one that meets the customer's desires for usefulness, initial cost, time of completion and total system life cycle cost.

This is a single sentence definition of what constitutes a successful system. I realize that attempting this is an activity fraught with dangers. Yet, I feel the necessity to try; given the definition's importance and the constraint of space. A definition of what defines success is another building block idea we need when addressing other tools and techniques.

This is a fundamentalists definition I hope. Notice especially that it alludes to no required "method" or "tool." The customer does not care how the system is develop so long as it meets the criteria outlined above.

Also note the emphasis on the importance of cost of the system over both initially and over its entire life time, the system life cycle.

Analysis and design need to be viewed in as integrated activities.

We need to see analysis and design as very closely coupled activities. One can find many examples and ideas within the professional literature and culture which separate these activities. It is, to me, incomprehensible how this separation came about.

I believe many problems associated with poor designs can be traced to the conceptual separation of these two tasks. I think it is unequivocally the case that analysis and design are inseparable activities if our goal is quality system designs. I will therefore use the term analysis-design from this point forward. I hope after thinking it about the reader does also.

The nature of software systems makes it difficult to achieve our definition of success.

Here we need to pause and state something for the record. Achieving successful software systems is not easy! Even achieving a moderately successful system usually takes a lot of effort and good degree of skill. I believe that we are constantly fighting against a form of mass brain-washing. This is in the form of numerous advertisements for various hardware and software companies and products that imply the answer are simple. "Buy this" or "do this" and all those problems your having with your business or systems will go away. We find ourselves dealing with people, users and management, who's knowledge is only this mass brain-washing or maybe knowing how to use Lotus 123. We find ourselves always feeling like we should be doing things faster, better and cheaper. Is it that all these intelligent and hard working people in the field have just been sluffing off? That is not the case. After years of doing and observing in this field, I believe that we need to acknowledge the intrinsic difficulties of creating successful software systems. We need to do this within the profession at least; in order to keep our attitude positive and our self esteem.

Software systems have many characteristics that we do not clearly understand. These characteristics are things like the fact that we are dealing with people and their thought processes. The possible ways that people can think about something is large. When one combines multiple thought patterns with the concrete problems in any software system the implications are complex and significant. Another characteristic has to do with change. If organization and users change then systems must also or they become less successful. Since organizations and users are in constant state of change... we are developing something which must actually be changing itself. This is the moving target syndrome. There are two examples of aspects to

software systems which we only barely have come to understand or acknowledge. We need to exert far more energy in understanding the many characteristics that are part of the "nature" of software systems.

Maybe by acknowledging and affirming the real and complex challenges of software systems we will be better armed in evaluating any simplistic solutions or answers.

Successful system design is more than "popularized" analysis-design methods address.

or

Successful top quality design is not simply a function of any method's diagrams and documentation!

We spent a little time laying out some ideas related to systems. We have an agreed upon goal of successful systems. We have acknowledged that we are dealing with something difficult. We have said that there are many complex aspects to this work which are less than fully understood. Is it any wonder that this profession's track record is a mixed one. We try very hard to achieve our customers expectations. What do we find? We find the ours and our customer's expectations are very hard to achieve in reality many times. One or more of the criteria is not met, in spite of our best efforts. This less than successful situation and our desire to achieve the best leads to a wish and even need to look for an "answer." In fact, human nature being as it is, we would especially like it if the answer was quick and easy!

In response to our desires the world seems to develop "answers." One answer offered for our desire and needs is that of "software engineering." The concept says we just need to apply "engineering" discipline to the system development process and we won't have these disappointing results. The challenge is that there are many vague definitions as to what "engineering" is in this context.

Not with standing the true lack of meaning to the term engineering, the term connotes a formal, clear, standardized, step by step process. Each step of the process will follow specific rules and be replicatable. This concept has a nice feeling about it. It implies a cleanliness and precision that is attractive to people who have "analytical" tendencies.

This enticing concept has given rise to a number of popularized (notice I didn't say popular or commonly used) "methodologies" that have claimed to define "software engineering." The Yourdon/Demarco and the, Warnier/Orr "methodologies" are two of the better known. A great deal of time and cost has been spent in learning and trying to implement these type of techniques. In fact these techniques have been around for well over ten years now.

Let's explore some aspects of what constitutes a complete quality system design and compare it to the popularized solutions. We need to focus on what a quality system design is because a design is the primary product that any analysis-design method or tools ultimately should be trying to produce. In studying the current concepts and "methods" related to system analysis-design I am struck by their narrowness of problem domain and consequently solutions. Let me elaborate. Most of the software engineering techniques would seem to indicate that the problem in system design consists only of figuring out what the database design should be or what "modules" need to be developed.

They pay only fleeting lip service to other issues. For example, problems such as human interface design and understanding the real system problem seem to at best get passing mention. Additionally, one almost gets the impression that we still can use single line hard copy Teletypes for input and output! To often it would appear that these methodological advocates are like little kids with a hammer...to which the whole world looks like a nail.

This is not to say there aren't differences in the quality of database designs. Or that there aren't better ways to design system modules. Yet, are these the most important and toughest challenges normally faced by designers in achieving quality systems. I doubt it. This focuses on the deeper issue. Namely, what are all the factors need to be considered when doing a system design. Are they merely the model of the underlying data structures and process flows? My belief is that all the popularized techniques have some benefit in some problem domains and some situations. But they are much too narrow in their conception of

what is required in order to development a quality software system.

In fact, watch what is happening to most of the popularized techniques. You'll see them trying to change to meet many of the real deficiencies people have found with them. The fact they are attempting to change is positive. Still, remember that they were sold in the recent past as solutions and answers with no deficiencies. Further, I have a feeling that if one could accurately measure the number of people who continue using the techniques today who have tried them in the past... we'd find only a very small percentage. This is I believe one of the best indicators of the true value of these so called solutions.

Further, I have seen little unquestionable evidence that any of these techniques has delivered on the promises of better, faster and cheaper. I'm sure their promoters might disagree with this assessment or give reasons why in a particular situation the promised results weren't achieved. What mainly exists is a great deal of testimonial "evidence" from the people who make their living selling the techniques! The bottom line is that they promise much more than they really deliver.

This leads me to an alternative belief. Our goal of achieving successful systems will not in the near future lend themselves applying any simplistic conceptions of "engineering" methodology. Rather, the elements that make for successful systems are many and more complex than anything being addressed in the current "engineering methods" and their "tools." In other words the process will remain a complex, "messy" and artistic process.

Software Engineering Requires "System Blueprints."

A fundamental deficiency brought out by a review of the system analysis-design arena is that we as system developers haven't spent much time defining what constitutes the deliverable of design specifications. Think about what design specifications means when describing a building or physical device. The blueprints of architectural or engineering design can be sent across the country to be built in two different towns. The building or device would be identical in almost every feature. What is the analog in system

design.... how many people design software-systems to this stringent criteria? Clearly, we need to develop a more universal and comprehensive set of design specification standards. This set of system "blueprints" is a precursor to developing any software engineering methods and the tools (i.e. CASE) to produce them.

Top quality system designs come from top quality system designers!

and

System designers/architects are not just "experienced" programmers!

We've been discussing the methods of analysis-design. Let's turn our attention to the people aspect. Think about the designers of physical things like houses or cars. Architects are not just experienced carpenters. And design engineers are not just experienced mechanics. They are trained in multiple disciplines all of which are required to accomplish the design of a physical objects or buildings. It is clear that systems are a much more difficult thing to design because they are not just physical objects. Given this fact, why do we seem to believe that the only required training for system designers is a few programming classes and a few years as a programmer?

Given this prevalent belief is it any wonder that the world finds our designs are less than successful many times. We will continue to have a lot of difficulties in this area until we recognize the difference between the skills, capabilities and training required of our programmers (carpenters, mechanics) and our system designers (architects, engineers). In fact, I believe, we need to develop a concept called the "system architect" to replace the old meaningless term system analyst.

Further the system architect must be conceived of as both the analyst and the designer. We must stop perpetuating the myth that excellent designs can be done by designers who are passed some model of the system on pieces of paper. The key to understanding why this is a myth comes back to realizing how few of the critical factors the current methods models document or take into consideration.

Additionally, in my experience the number of excellent system designers is some very small fraction of all system people. We need to figure out how to identify these individuals. This is tough task but essential if we are to meet our goals. Given a choice between an mediocre system architect with "all" the methodologies and an excellent one with "no" methodologies, choose the latter!

Conclusions about system analysis-design.

What conclusions can we draw from our discussions about system analysis-design. First, we have a great deal more to learn about what software systems "are" and what software engineering them might entail. Second, the "solutions" that have been popularized in the last ten years have only limited utility. Third, quality system analysis-design is very much a people oriented issue. Lastly, we will need to keep our minds open to the complexity of what we do and therefore the complexity of it's solutions.

Case - Computer Assisted Software Engineering

We've reviewed some key ideas about system design. This gives us the basis to now evaluate CASE.

What is CASE - Computer Assisted Software Engineering?

The name CASE would appear to be a self defining acronym. In other words, does a tool assist in software engineering...then it must be a CASE tool. But as we've seen, the term software engineering has many meanings... mostly revolving around a few popularized methodologies. And in fact, there are many products out there that claim to be CASE tools that differ from each other to significant degrees. These facts clearly indicate that the definition of what constitutes CASE is pretty broad and loose.

The fact that no definitive definition for what CASE exists should put us on guard. Just because someone says a product is a CASE tool means nothing in particular. This situation alone should cause one to have a cautious attitude about CASE. Armed with such an attitude we can now address the concept of CASE in a realistic light.

CASE Tool Features

Let's list some of the issues and functions for which the CASE tools try to provide "computer assistance." Note that the great majority of case tools address just a subset of this list of areas. Here is a composite list of areas addressed by CASE tools.

1) Computer Assisted Diagramming Tool(s) - for one or more of the following analysis methods:
   - Entity-Relationship Diagramming
   - Data Modeling
   - Data Flow Diagramming - Yourdon/Demarco,
     - Gane/Sarson, etc
   - Real Time Process Diagramming
   - State Transition Diagramming
   - Warnier-Orr Diagramming
   - Flow Charting

2) Assisted "Method" Verification

3) Database Analysis-Design Support

4) Computer Assisted Diagramming tools for Design "methods".

5) Data Dictionary - Repository for information about the system.

6) Report and Screen Format Layout

7) Module Code Generator

8) Reverse Engineering of Existing Systems

9) System User Documentation

Warning! Once again it is important to note that no product provides the functionality in all of these or even most of these areas in the way you would really like. CASE vendors appear to be foremost marketers. They sling the buzz words associated with our real needs, desires and problems... with easy abandon. Unfortunately, the implied promises appear much faster and easier than delivered results.

The allure of CASE.

The major reason for the visibility of CASE extends back to our discussion of analysis, design and programming method "solutions." We are looking for an answer, maybe even an "easy" answer to our very real challenges. Case is alluring because of what it promises... assurances of correctness from the 'engineering' and ease from the 'computer aided.'

We live in a free enterprise system where people and companies can make a product and service and try to sell it to us. CASE tools are an attempt to answer a real need. Our challenge is to not be overcome by the combination of our needs and the CASE vendors promises.

CASE tools are build on a foundation open to questions.

All the CASE tools I've reviewed rest squarely on only a few analysis-design "methods" which have been advertised as "solutions" in the last few years. We discussed these previously when looking at system design. These methods are the "software engineering" for which CASE tools are attempting to provide computer assistance. Here is THE question...if the underlying "methods" are really only a very small part of what will someday be called software engineering then can these tools be anything but partial solutions? The bottom line is that the CASE tools I've seen are too simplistic in their conception of what constitutes system analysis-design. Consequently, they may be useful for dealing with certain aspects of certain problems in certain situations.

In fact the CASE tool consciously or unconsciously reinforces that belief that software engineering can be a "one, two, three" process which can be automated. As I stated earlier, we must understand the importance of the human factor. We are a long ways from the day that any methodological model.. assisted, automated or not, will result in quality design solutions.

Who will benefit from CASE tools.

Not withstanding the limited uses of the methods that CASE technologies support; those people who are already actively using the "methods" that a particular CASE tool focuses on will stand to benefit most from

using the tool. CASE tools will not by themselves suddenly make systems people knowledgeable in any of the "methods." In fact, my experience is that CASE will just much more rapidly pinpoint the lack of knowledge and practical skill level that a person has in regards to a particular method.

CASE will benefit those who can afford to utilize one of the "methods." My experience is that these are large shops in business and government. They seem to be able to "afford" to spend great amounts of time jumping through the methodological hoops. It is not clear that small shops will be able, in the near term, to afford the added overhead of any of the "methods." This is true whether done manually or supported by CASE tools. Rather these shops should be focusing more on using what Tom Lister calls BASE - Brain Assisted Software Engineering.

Case tools need to handle the entire System Life Cycle (SLC).

This idea was touched upon with other focuses earlier. Here I want to highlight the limitations of CASE having to do with it not dealing with the entire system life cycle. My belief is that if CASE tools and the methods they support are to be the most valuable then they need to be an integrated part of the entire SLC. Some of the more expensive CASE tools claim to actually deal with the entire SLC. One should approach these with a Missourian attitude of believing when you see it and use it. The bottomline is that most of the CASE tools focus on only a part of the system development phase of the life cycle and therefore significantly limit their value.

Summary of CASE

CASE tools imply and promise more than they can deliver. They are built upon a shaky methodological and system view foundation. That is not to say that they won't be helpful in doing certain system development tasks, either faster or better. They will not however make a "software engineering" miracle happen any more than a word processor writes a excellent novel. With this knowledge we can sensibly make decisions about evaluating, obtaining and, at this stage, experimenting with CASE technology.

Object-oriented programming.

If the buzz word for the 70's was "structured" and the buzz word for the 80's "relational" or "software engineering" then that for the early 90's is "object-oriented." We'll see "object-oriented" everything... languages, databases, analysis-design methodologies, etc. What is this latest craze all about? Will it be the answer any more than structured, relational or software engineering has been in the past? Let's take a look.

What are the fundamentals of "object-oriented?"

Here are the key concepts underlying the object-oriented model and strategy.

1) Objects classes - which resemble the "entity" concept in data modeling.

2) Objects contain both data and the methods to access and process the data.

3) Data processing implementation and data structures of objects are hidden from "outside" world.

4) Inheritance - Object classes can be "derived" from other object classes. Inheriting their data attributes and methods.

5) Polymorphism - a method defined for a parent object class can be redefined to meet the needs of the child class.

6) Reusability - fundamentally a concept along classic lines of the reuse of data structures and methods.

The above are the general principles most often associated with object oriented approach. They are in some ways an evolutionary rather than a revolutionary step. This approach emphasizes and combines strategies used in the past and new ideas along with languages to support them. So if you are caught by the thought... we've done things like this or that before...you're partially correct.

The definition of object-oriented is not absolutely defined.

It is important to understand the uncertainties associated with the definition of object-oriented as of this date. As with CASE, there are numerous products and pundits all claiming to have "it" or know what "it" is! This state of affairs sounds so similar to that of CASE and analysis-design methods. This similarity should raise the caution flag when looking at any object-oriented approach.

The definitional variety is very evident when one looks at all the different languages that claim to be object-oriented. There are definitely similarities between them and yet there are significant differences.

Anyone looking at using the object-oriented languages is advised to understand the differences and their implications.

Object-oriented programming will require new methods of analysis and design.

If one tries to use the object-oriented programming languages that exist using the analysis-design mindset of procedural languages...problem rapidly arise. It is generally recognized that new methods of analysis-design will be required to be developed, learned and applied before we will see many of the benefits of object-oriented programming itself.

Given this requirement, my review of the latest thinking in the area of object oriented analysis-design does not indicate a quick solution to the problem. There are numerous people trying to come up with how you do this object-oriented analysis-design. And you will see a spate of books on the subject within the next year. In spite of this, I believe you will find that the techniques are less than the solutions we will need. This state of affairs will diminish this approaches usefulness. Further, it will also limit our testing and assessment of the validity of the object-oriented model in the near future.

Object-oriented will require higher level of competence and skills.

The fact that higher level of competence and skill will be require for object oriented contradicts some of what you may be hearing. Many proponents allude to the "fact" that object-oriented programming will be much more productive than traditional means. Productive however may not mean easier. It sure is the case that the design of what objects and their methods are more critical in the object-oriented arena than in more traditional ones. It would also appear that the skills required to design and develop systems within this environment are greater, at least, for some tasks.

The best information available would indicate that as long as six months may be required to train a person to think and program within the object-oriented paradigm. Since we are all experienced with switching to new languages much more rapidly than that.. it is a definite indicator of the difficulties, of possibly a new nature, in implementing this new approach. The true meaning of these learning difficulties has yet been revealed.

The CASE tools in existence will all claim to support object-oriented analysis-design.

We have already indicated that there is still much uncertainty about what object-oriented programming and analysis-design is exactly. It can be taken as pure marketing hype therefore if you see any CASE tool claiming to be supporting object-oriented in the next year.

No object-oriented language exists currently in the HP3000/MPE environment.

Object-oriented languages will appear for the HP3000. But currently none exist, at least for the MPE operating system. The first language we are likely to see is C++ since this is a language which is translated by a pre-compiler into C.

Some features of the object-oriented model can be approximated in current languages. The sub-program in the COBOL environment allows for achieving some features such as method and data hiding. Pascal allows

for one to inherit data structures from another data structure. Message files are a way of passing messages between processes. Yet, languages which directly support the object-oriented concepts definitely make implementing this approach much easier.

Performance will be a significant issue for some applications.

It is the case that there is a computer performance price to pay for using object-oriented approaches and languages. Whether it is significant will have to do with application design tradeoffs, particular languages and hardware.

The argument that is brought forth to counter this downside is that we will make it up in more reliable, flexible and less expensive systems to build and maintain. This will not be an acceptable tradeoff for all applications.

Conclusions about object-oriented programming.

I have indicated some uncertainty about the nature of the object-oriented approach. This is definitely the state of the art of object-oriented... uncertainty. Nevertheless, I believe the concepts underlying the object model will prove to be significant!

Our professional task is to weather the hype in the short term while applying what is truly valuable. The hype and promises will get very fast and wild in the near future. One is forewarned that these are mostly marketing hype and not "bankable" useful techniques or tools.

At the same time, the object-oriented approach will prove to be a useful new model. It will have benefits in certain problem domains. It is less clear and early indications are that it will not be the answer in every problem domain!

What is needed are some concrete examples, within the problem domains of common HP3000 applications, of successful use of any object-oriented model and its tools. Pioneers, westward OO (Object-Oriented)!

The "State of the art assessment" of the "State of the art."

We've covered a lot of ground in looking at design, CASE and OOP. I've stated my conclusions in each area and won't repeat those particular ones here. There are some overall conclusions that I believe can be drawn. First, we, in software systems, are involved with a challenging, multifaceted and moving problem. Second, the "nature" of the problem is far from understood. Third, it may be that software systems have a "nature" that doesn't fit into any known conceptual models, for example "engineering." Fourth, in the foreseeable future, achieving quality systems will be dependent mainly on the brains of the people doing them. Finally, there are no easy answers! In spite of this fact there are days I'd give a years pay for one!

The Author

Rob Mattson is currently Manager of Information Systems at WIDCO, a large open pit coal mine in the state of Washington. He has been working in the systems field for 14 plus years, the last 11 in the HP3000 environment. His background in addition to many systems related roles includes a degree in Psychology, work as a Certified Public Accountant and as a management consultant/trainer. His current professional areas of interest include general systems theory, project management and exploring the concept of system quality. When not immersed in systems he enjoys his family, sailing, sea kayaking and golf.

Paper Presented Originally at Interex Computer Management Symposium, Las Vegas, Nevada March 1990.

A Checklist For Managing Today's Data Centers: Have You Forgotton Something

Betsy Leight
Operations Control Systems
560 San Antonio Road
Palo Alto, California

INTRODUCTION

The efficiency and security of data processing operations is a major concern to corporate data processing managers.

To assess their data center's performance, management often conducts a review of the following areas: standards and procedures, operational work flow and control, scheduling, data security and access control, equipment utilization and environment.

If the data center management and staff understand the concerns of upper management and the information that is needed, the operations review can proceed smoothly, and the results can be beneficial to the entire organization.

Corporate direction for data center management focuses on control issues, automation procedures and the gray area between. The data center manager reviews the operations and interprets corporate objectives to the operations staff. Because this article focuses on the concerns often overlooked by the data center manager, it can be a useful check list for improving daily operations as well as preparing for a data center operations review.

STANDARDS AND PROCEDURES

Data center managers should verify that standards and procedures exist and are enforced. These written rules are the controls; they should include:

* Ensure proper timing in running programs and jobstreams.
* Insert changes into production runs; enter run dates.
* Use correct data for programs; access correct data files.
* Protect data and programs from accidental or intentional destruction.
* Specify methods of physically moving input and output.
* Schedule work and getting work rerun in the event of errors.
* Keep records of work performed and session logons.
* Determine and record sufficient resources for the work.
* Perform maintenance and general housekeeping associated with the operation of the data center.

The data center manager should ensure that formal standards exist for systems development and maintenance, program and system testing, file conversion, program and system change control, library operations, computer operations and documentation.

For each aspect of standards and procedures, your installation can implement procedures using automation software as shown in the following chart.

Task Description	Controlled Operation	Automated Controls	Controlled Automation	Automated Operation
Run Programs	User Task	User with Scheduler	User who Schedules	Scheduling Software
Change Jobstreams	User Task	User with Editor	User who Edits Jobs	Job Change Software
Verify Data File	User Task	User with File Scan	User who Scans Files	File Scan Software
Monitor Jobs	User Task	User with Jobstream Monitor	User who Monitors Jobstreams	Jobstream Monitor Software
Control Master Files	User Task	User with File Copy	User who Copies files	File XFER Software

OPERATIONAL WORK FLOW AND CONTROLS

The data center manager should investigate specific items in this area, including whether:

* Enter input data from other departments completely and on time.
* Track job accounting and session logon information.
* Notify appropriate personnel in case of production processing error.
* Document batch processing errors and logon violations.
* Accumulate error statistics.
* Follow up on all errors so they do not recur.
* Distribute all reports to the proper user departments.
* Establish procedures to control the distribution of sensitive output.
* Dispose of confidential reports when they are no longer required.

The data center manager should also confirm that downtime is reported and statistics compiled. A log of late reports and jobs should be maintained.

There should be a formal communications channel between data center operations and other departments; operational tips and other advice should be passed to all operators.

All problems encountered at the computer, as well as any action taken to prevent their recurrence, must be documented. Operators must also receive feedback on reported problems.

The data center manager scrutinizes output report distribution and disposal and determines whether:

* All reports have been distributed to the proper user departments.
* Procedures exist to control the distribution of sensitive output.
* Procedures exist for disposing of dated confidential reports.

Finally, the data center manager should ensure that jobstream run instructions are kept up to date.

SCHEDULING

Efficient and effective scheduling is extremely important in providing a high level of reliability and predictability to data center operations. The data processing manager should determine whether:

* Daily processing activities are scheduled and a daily contingency schedule is maintained.
* Actual run times are recorded for batch programs and jobstreams.
* Data is used to calculate expected run times for a given day.
* Expected run times are compared with actual execution time to ensure that processes have not been terminated abnormally.
* Unscheduled runs are supported by a work request or other written authorization. Schedule deviations should be documented and followed up on by a supervisor.
* User-submitted jobs are recorded to allow forecasting of future schedules, resource requirements, and special processing considerations.
* All jobs are submitted through or controlled by data center operations.

Scheduling software enforces controls and totally automates the data center operations. Manpower reductions can result depending on implementation. A chart on scheduling appears below:

Task Description	Controlled Operations	Automated Controls	Controlled Automation	Automated Operation
Create Schedule	User Task	User with Streamer	User who Schedules	Scheduler Software
Monitor Run Times	User Task	User with Job Log	User who Logs Jobs	Monitor Software
Job History Report	User Task	User with Job Log Data Base	User who Compares Job Logs	Data Base Report Generator

DATA SECURITY AND ACCESS CONTROL

Data base and master file information should be protected from unauthorized access or loss. Employees must be instructed about their responsibilities concerning confidential information. Management should periodically review and update controls and security provisions relating to data.

Live production programs should be physically separated from development programs. The staff should be prohibited from running test programs against live files, and operations personnel should be denied access to sensitive data files.

Secured file management is not limited to source and object control. Data center managers should ensure that procedures have been established for:

* Transferring programs from development to production.
* Approving program library changes.
* Testing changes to programs before transference to the production libraries.
* Updating production documentation after changes.

To maintain security, operators should be prohibited from renaming or transferring programs without supervisory approval. Internal labels must be used from all data and program files.

Passwords and lockwords should be used to protect accounts, users, data files and port access. Passwords, lockwords, dates and constants should be introduced at run time, eliminating the need to hard-code sensitive data in jobstreams.

Data security and access control software can bring automation to the data center; with automation you can expect a more efficient system operation as summarized in the chart below.

Task Description	Automated Controls	Controlled Automation	Automated Operation
Separate development and production areas	User who moves files	User with file mover	File Librarian Software
Restrict live file access	User who locks files	User with lockwords	Protected File sets to User Sets
Approval pre-step to Production Move	User who moves files	User with file mover	Automated File move after Approval
Project/memo notes Related to Changes	User who completes forms	User with text editor software	Online dialogue requesting memo text at save time

EQUIPMENT UTILIZATION AND EFFICIENCY

Once it has been determined that the entire data processing department is following a properly implemented set of standards and procedures, the data center manager should review equipment utilization.

The data center manager should collect raw data from the system log files in order to report the following information:

- How much machine time is spent on reruns?
- How can the reruns be best analyzed?
- Which jobs are especially susceptible to reruns?

With reported resource utilization information, the data center manager should check that the full multiprogramming capability of the system is being used. It then follows that multiple jobstreams should run concurrently, if there are no data file bottlenecks.

The data center manager then reviews whether many jobs can be restarted without rerunning the entire job. Jobstep tracking and restart software should be implemented for efficient data center operations.

ENVIRONMENT

The data center manager should review the work space to ensure that it is adequate for the number of employees. The environment should be neat, and supplies should be easy to locate.

Auxiliary items located outside the computer room, such as bursters and de-collators, should be accessible for the flow of work in the department. Tapes, discs and other storage media should be stored in a closed, fire-protected, limited-access area.

RECOMMENDED COURSE OF ACTION

The data center manager should make the organization aware that the following steps can enhance the operations review:

* Providing management with as much information as possible.
* Implementing software systems that leave clearly defined audit trails.
* Keeping accurate records, log files and file history information
* Maintaining formal written standards and procedures.
* Implementing an effective data security and access control system.

Following the data center manager's recommendations and procedures for operations can yield an efficient, secure and automated data center.

TITLE: Computer Aided Project Management

AUTHOR: Lewis R. Dalrymple (602)731-8466
City of Tempe
120 E. 5th Street
Tempe, AZ 85281

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 3004

Effectively Managing Your System's Resources... Other Avenues

by Carl M. Rusk
Infocentre Corporation
2710 Stemmons Frwy. Suite 1100
Dallas, TX 75207
(214) 631-1976

When it comes to managing your company's resources, MIS at times, needs to search for other means for capturing system resources.

When talking about computer resources, it usually comes down to CPU performance and "Disk I/O's." There is the standard fare of products that will help in both of these areas. Tools such as "MR NOBUFF" readers, high-speed relational extraction tools and hardware enhancements are just a few on the list. For MIS this may not be enough, you may need to take advantage of Other Avenues which may not have been explored before.

There are three areas that come to mind that are often overlooked: 1) Controlling your end-user reporting environment; 2) Taking advantage of your corporation's MIPs and disk space already accumulated in the form of PC's; 3) And finally, getting the biggest return on investment from your MIS systems development staff.

This paper will explore these three areas of data processing in more detail so you can take advantage of some resources that you may have overlooked.

END USER COMPUTING:

Does the thought of giving an end-user reportwriter to a bunch of unsophisticated end-users conjure up visions of doom? Usually. As a DP shop you have to decide whether on the one hand it's an advantage to have end-users do their own reporting so some of your time is freed up for more complex tasks. On the other hand the users might abuse the privilege and bring the CPU to its knees which may cause harsh feelings between the users and DP. What happens then, do you take the privilege away? This of course would put the reporting load back in the hands of DP, or do you continue allowing the end-user do their own reports knowing very well that the CPU could be buried at any time. Sounds like a "Catch 22."

The Hidden Resource:

Even though we cringe at the thought of letting end-users loose with a tool like a reportwriter, the end benefits

usually out weigh the penalties. You can think of your end-users as a hidden resource in a manner of speaking. They are getting the information they need themselves without putting too heavy of a burden on DP.

The Goals:

To implement an end-user tool like a reportwriter you will need to set up some criteria for selection. Some of the areas to look for are:

Ease of Use: If the product is too difficult for the user to navigate then more than likely it will be destined to failure. Look for a reportwriter that offers a lot of online help, menus, function keys, etc.

Flexibility: If the product is simple to use but the user can't do much with the product, then guess what? The user is coming back to you (DP) for those reports.

Automatic Links Into Other 3rd Party Tools: As we all know, 3rd party software vendors can offer a wealth of tools to you to take advantage of. If the reportwriter you're either using or evaluating won't interface to them, especially automatically, then you will be forced

to write some interface (if the product will allow you to) or worse yet you will have to purchase one!

Control: This by far is the most important of them all. If DP has the ability to control who can run which report when and who has access to what part of the reportwriter, then you're really in control of that end-user resource.

You need to evaluate your particular environment and ask yourself some questions to determine if End User Computing is for you or your shop.

PC INTEGRATION:

In most corporations today you will find a PC or two doing specific tasks; such as word processing, spreadsheets, graphics etc.. When you talk about PC's and HP 3000 computers in the same sentence it usually means terminal emulation. That's almost an insult to the PC in a way. The thought of using this small processor and disk drives for something as mundane like being a dumb ASCII terminal is a waste. Well, all is not lost, vendors have come to the realization that, PC's are like little computers.

The Numbers:

Have you checked your PC count lately? If not, you may be surprised to learn that PC's are slowly replacing terminals

for user workstations. There are two primary reasons for this, one is that the price of a PC has dropped significantly putting it into the same dollar arena as the terminals. Secondly the amount of software that is available on the PC can help the user become very productive.

Well, if all these PC's are showing up on all of these desks, then let's convert these PC's into some meaningful numbers. If your company has 30 or so PC's with 640K of memory each and 40 Mb of hard disk each this would give you 18 Mb of main memory and 1,200 Mb of disk storage!

The best part about these figures is that even though you have essentially the power of a series 70 in those PC's, each unit is an independent processor. This means each user isn't contending for CPU resources with everyone else.

The Software Catch:

With all this CPU and Disk available it almost sounds too good to be true, right? The real trick is to find some way of integrating all of those PC's to an HP 3000 to make it all a reality. There are a lot of old and new **"BUZZ"** words

out there like **"Distributed Processing," "Cooperative Processing,"** etc. They all pretty much mean the same, run the application on the PC and share the data from the HP 3000.

To do this one needs a common data structure, a common language, along with some sort of file server that would reside on the HP 3000. In this type of environment some of the items you will want to be aware of are:

Common Data Structures: If you are using IMAGE or a combination of IMAGE with a relational type enhancement like Omnidex or Superdex on the HP 3000. Then you will want to find something close to that arrangement on the Personal Computer. If you don't you will need to have a Data Base Administrator that knows IMAGE on the 3000 and whatever on the Personal Computer. Worse yet you may have to hire a new employee that knows the data structure on the PC.

Common Development Languages: If you are planning to move an application from the HP 3000 to the PC. It will become less painful if the languages are the same. Fourth Generation Languages will give you better flexibility in this area because they have their own screen drivers etc. (View 3000 won't run on the PC)

Common User Interface: This portion sometimes become very crucial for the acceptance of a system. Let's say, for example, that you develop a system on the HP 3000 in COBOL and View. Now you replicate the system on the PC using some type of PC tool or language. Even though they accomplish the same given task, they will behave and look differently. Try to explain to the user that they have to learn two different versions of the application to get their work done. This can be very frustrating for an end user.

Data Transfer Capabilities: Batching files on groups of data up to the HP 3000 or back to the Personal Computer are fine in some cases. In your shop that idea may not be indicative of real life. To take full advantage of those PC's and your HP 3000 in a realtime interactive environment, you will need to seek a product that can pass data interactively back and forth. This may be based on demand or some preset time limits (time stamping).

One has to be careful though, if you find a vendor that can offer this type of environment (interactive or cooperative processing) don't get caught up in the flexibility of the software and kill the application. As an example, if you set up the application to continually read the data files on the HP 3000 then you would be better off just using terminals. On the other extreme the application may be interrogating the data on the HP 3000 maybe just once a month. This would be a good indication that the application wouldn't need an interactive environment to the HP 3000

data. You have to gaze into that crystal ball and really evaluate your application.

If you seek the right tools and implement a sound application then the amount of resources you will acquire from those PC's will be incredible!

APPLICATION DEVELOPMENT:

What is the most valuable and by far the most perishable resource in your Data Processing Shop? Without question **TIME**. And it's funny how every task, whether at work or home, takes valuable time. So what does one do to preserve some of that valuable resource? Work more effectively and efficiently.

Finding The Time Bandits:

Time bandits are as the name implies and they come in many forms. The people who are affected the most by these bandits are you, as well as your entire Data Processing organization. Let's try to identify these time bandits and then formulate a plan to help make your shop more efficient.

Program Backlogs: In almost every data processing shop there is a backlog. They can take on three primary forms. Programming backlogs are by far the largest and the most costly of our time bandits.

The first is new development or new request backlogs and depending on the company this backlog could be very large or hardly insignificant. If your company is suffering from a new development backlog this could be a tremendous load on data processing since the request has to be analyzed and laid out, then actually programmed. This process can take months, even years, depending on the size of the project. After it's implemented the user will always come back and say **"this isn't what I asked for....!"**

The second type of backlog is your typical maintenance request and this could be the most dreaded and the most painful. The reason is, most of the programs currently in production are older, and written in COBOL, FORTRAN, etc. by someone who is no longer with the company or who never documented their code. Our friends, the time bandits, are very active taxing your time on these projects. Not only are they time consuming for the actual coding but the amount of time it may take to find the affected code let alone trying to figure out what the program is doing can turn into a tedious task.

The worst part of any backlog is what I refer to as **"the hidden backlog."** The hidden backlog comes about when the visible backlogs become so big, the users figure by the time data processing gets around to my new request I'll be in retirement. The user may have written off the request, but it is still there in the back of their mind waiting to spring to life. Even if you start to catch up on your normal backlog and think you're ahead of the game, suddenly the user realizes that all is not lost. Now the user may feel it's a good time to submit their request once again.

The Backlog Solution: The answer to the backlog problems can be many, but be forewarned that none of the answers will come cheap to a data processing shop or company. Keep in mind, that any major purchase be it either staff, software etc. can usually be justified over time and depending on the solution you choose. Your **Return on Investment** could conceivably happen within a few months following the purchase.

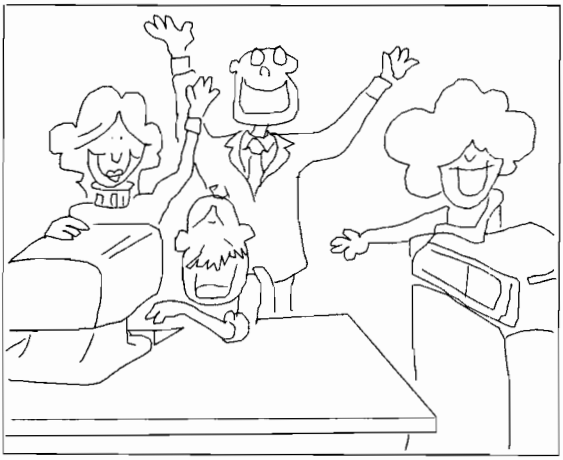
There are two possible ways to help cut programming backlog that I'm aware of and it's entirely up to you and your environment which might fit best. The first of course is to throw more resources at the problem or quite simply add programming staff either permanently or through a consulting relationship. Adding a new member to your staff will be harder to justify then hiring a consultant. A full time person means a continual outlay of salary plus medical and retirement expenses. A consultant is more like a temporary in that you can get the help for as long or as short of a time as required.

The second alternative to your dilemma would be to implement programming aids such as code generators or 4GL's. Now I know what you're thinking, but most of the software vendors out there have been listening to the masses over the years. As a result, the 4GL's have become more flexible along with being performance conscious. If you take a look at the numbers, the price of a 4GL will equal the cost of a full time employee (1 year's salary) but will typically give you the programming productivity of 5 programmers! Your 4GL vendor should be able to provide you with these numbers as well as cost justification numbers to help you sell the idea to your management.

The CPU Crunch: Another time bandit that may have been overlooked in the past is CPU performance. It will affect not only the users on the production system but it can severely impact the work effectiveness of your data processing staff. Consider a programmers work cycle of using a typical editor to modify their code, save the text file, compile the program and finally test their results. You can see how this repetitive flow can turn into a very time consuming process if your current CPU is at it's limits.

The CPU Solution: Once again the two possible solutions for this particular problem could have an expensive price tag associated to it. The possible savings over a year or two could justify the expenditure. One additional note, this is not to say that these solutions are the only answer but they are the two I have arrived at.

The first option would be to look at upgrading your current CPU. This of course would require a lot of planning and financial resources. The overall benefit to all users in the company as well as data processing would be a great shot in the arm. One item to keep in mind when looking to upgrade your CPU is to gaze, once again, into your crystal ball and try to formulate the growth of the company. Then you can anticipate how large the CPU needs to be on that given year. The worst that can happen is you go to management and get the funds to upgrade a series 52 to a 70 and a year after the upgrade your computer is on it's knees again. By then I hope Your resume has been upgraded.



"... The budgets been approved for the 950!"

The second option is to consider purchasing a small 3000 like a Micro LX or GX. These baby 3000's pack a pretty good punch for their size and cost is in the area of a well outfitted PC. The other added advantage is that they don't require any special wiring or air conditioning because they're an office environment computer. If your staff is larger than say 5 to 6 programmers, then consider maybe more than one keeping a limit of about 6 programmers per CPU. You may just want to consider going up to a 925 LX that essentially is a full blown 925 limited only by the operating system at a much more attractive price tag. Once again the 925 LX is an office environment computer not requiring any special wiring or cooling. The second alternative may not be as costly or take as much planning. I would

highly recommend any data processing shop to consider this option especially if your current CPU is groaning under the load.

We have identified some of the dreaded time bandits that might be lingering in your data processing shop. I have given you some ideas to help you evolve your programming staff and DP into an efficient entity.

That's a Wrap:

Systems resources is a fairly broad term, I only hope that the information that we have mentioned here might be helpful to you in some way. At least, I hope it may have spawned some alternative ideas that will start you on your way to a more successful data processing shop.

EXPERT SYSTEMS: YOUR STRATEGIC DECISION

Prepared for the 1990 INTEREX ICMS, Las Vegas

Ross G. Hopmans
Brant Technologies Inc.
2605 Skymark Avenue
Mississauga, Ontario
CANADA L4W 4L5
(416) 238-9790

INTRODUCTION

The premise behind this paper is that people involved in computing management approach application development as a problem solving exercise. We begin with a problem and we must determine the most appropriate way to solve that problem. This paper and the tutorial for which it was developed are meant to introduce expert systems technology as a problem-solving technique to an audience of managers so that they can make intelligent, strategic technology decisions.

We will investigate what expert systems are and examine how some organizations are using them. From there we will discuss where expert systems really fit - what are their advantages and disadvantages and how can we recognize when expert systems are appropriate to an application. Finally we will examine the nature of an expert system project and some of the software choices for expert systems development. Finally we will draw conclusions about how this technology can integrate with other forms of information technology.

WHAT ARE EXPERT SYSTEMS?

Expert Systems are a branch of the larger field of Artificial Intelligence (AI). The goal of AI has always been to produce intelligent computer systems. Let's start by looking at some of the differences between Artificial Intelligence and Biological Intelligence.

Biological Intelligence	Artificial Intelligence
▶ perishable	▶ permanent
▶ difficult to transfer	▶ easy to duplicate
▶ erratic	▶ consistent

▶ difficult to reproduce	▶ easy to document
▶ creative	▶ uninspired
▶ learned	▶ programmed
▶ sensory input	▶ symbolic input
▶ wide context	▶ narrow focus

The term expert system describes an important technical issue: it indicates that the source of the program's power is primarily a large body of task-specific knowledge. Such systems may function in a variety of roles acting as assistants, colleagues and sometimes as experts. They can do that because the technology underlying expert systems allows us to do symbolic inferencing and not just arithmetic. It turns out that most of the interesting things we know about the world are not numeric and most of the knowledge we have about the world is not well modelled with arithmetic. We think and reason about problems using simple IF/THEN rules and expert systems allow us to easily handle this type of logic.

Most of the tools and techniques used today to develop computer applications are oriented toward numbers and algorithms. There is no denying the power of those tools or the power of the computer systems in use. However, computers have been used primarily to do things that people are not very good at. Things like filing, sorting and processing large volumes of data quickly. Expert systems fill a different role. They are used to automate processes that people are good at. Things like examining alternatives, making recommendations, giving advice.

An expert system is a close approximation to cloning. What we cannot do biologically, we try to do intellectually. We take some rare and valuable expertise and try to clone it by talking at length with the person who has it. It is useful because almost all organizations have a knowledge bottleneck - a place where its productivity is limited by a scarcity of knowledge and skill. Whom do you miss when they are sick? Whom would you hate to lose to the competition? Whose impending retirement has you worried? Whom do you wish you had five more of? Cloning that knowledge is one possible remedy to these problems and expert systems offer us this possibility. The focus of expert systems is on knowledge - collecting knowledge, formalizing knowledge and putting knowledge to work in the computer.

Expert systems take a particular view of the answer to the question "Why are experts experts?" Do they think faster than the rest of us? Do they think differently? Do they have a general purpose trick of thinking or problem solving? To call something an expert system subscribes to the belief that experts are experts because of what they know. Expertise arises from knowledge. The power of expert systems comes from their base of knowledge about the specific task at hand.

Expert systems technology is nothing more than a collection of tools and techniques that allow us to solve problems that can be expressed as a series of IF-THEN rules. A few hundred to a few thousand rules provides us with a significant base of knowledge. An expert system can be developed with any language or tool although, as we shall see, there are tools and languages which are specially designed for these types of systems. The point is that an expert system is identified by the programming technique that is used.

Expert systems are characterized by their reliance on large stores of rule-based knowledge as a basis of their expertise. In effect, we have done a dicing of the human expert's knowledge into individual rules. Rules are comprehensible and each stands on its own feet. Each rule makes one small decision that relays one thing to do under a particular set of circumstances. We use a rule whenever it is relevant - that is, whenever its pre-conditions (the items in the IF part of the rule) have been met. In traditional programming, we write complete decision-making procedures, whereas with expert systems, we write individual decision rules. Traditional programming tells the computer what to do; here, we tell it what to know.

Frankly, it is very difficult to express complex logic using the tools and techniques of the typical business environment. Complex logic is poorly understood, rarely documented, difficult to implement, time consuming to test and impossible to maintain - particularly if we consider that decision making logic is dynamic and must be able to change when policies or economic circumstances change.

This is the primary strength of expert systems. Expert systems technology provides us with tools and techniques designed to implement and maintain complex logic. People think and reason in terms of IF-THEN rules. But the conventional data processing languages and tools do not provide us with a model of human problem solving. As powerful as they are, they were designed by engineers from the point of view of providing instructions to the computer that were most convenient for the computer to use. Expert systems technology and logic programming take a different approach. They provide a model of human problem solving and work from the premise that languages and tools should accept instructions that are easy for people to provide. They accept rules and they turn the computer into an engine whose purpose is rule retrieval and application.

Expert systems are often called knowledge systems or knowledge based systems because they operate on a knowledge base of facts and rules. The knowledge base is the heart of an expert system and so the emphasis is in the development of that knowledge base. Development of the knowledge base is part of the knowledge engineering process.

HOW ARE EXPERT SYSTEMS BEING USED COMMERCIALLY?

Before we continue our exploration of what expert systems are, let's take a look at how some expert systems are being used commercially and make some generalizations about their strengths.

Edward Feigenbaum of Stanford University documented some of the dozens of companies he investigated world-wide that are using expert systems in his new book, The Rise of the Expert Company (Times Books, 1988). He discovered expert systems successfully carrying out tasks of inhuman complexity: permanently capturing expertise in a form that won't retire or change jobs; significantly improving the quality of work; transforming hard jobs into easier ones; and improving knowledge worker productivity in a host of other ways.

The key question for management is "Are expert systems paying off?" and the answer is an emphatic "Yes"! Almost everywhere they are being used, expert systems are speeding up their segment of knowledge work by at least a factor of ten and speed-up factors of 20 and 30 are common. We cannot underestimate the impact of a 10X speed-up. A 10X effect is the difference in speed between walking and driving, and between driving and jet plane travel. Engineers usually reserve the term "revolutionary" for a 10X effect.

The expert system examples presented here were described at the First Annual Conference on Innovative Applications of Artificial Intelligence last year in Palo Alto.

Manufacturers Hanover Trust Company developed a system called TARA (Technical Analysis and Reasoning Assistant). TARA is a real-time system which monitors an international data feed in currency trading and provides traders with recommendations on buying, selling or holding market position. Currency trading is a high risk, high reward activity and any small increase in the accuracy of predictions will lead to significant payback. Large volumes of data can now be analyzed in real-time with consistent expertise. In less than a year of operation, TARA has paid back all of its development costs with an attractive return on investment. TARA provides a competitive edge for MHT and it has had a widespread effect on the organization's attitude toward technological innovation.

GMAC has developed the Advisor for Financial Analysis of Automobile Dealerships (ANALYST) to assist the 230 locations in which it has been deployed in performing regular credit risk analysis on automobile dealerships. It reviews financial statements submitted by the dealership, predicts the dealership's likely performance until the next scheduled review, recommends credit lines and suggests ways to reduce risk. Tangible benefits include a 50%

reduction in time to perform the reviews, reduced losses due to bad risk assessment, elimination of training, and reduced effort for compliance audits. Total payoff from these benefits is estimated at $2 million per year. Intangible benefits include standardization of the review process.

One of the more highly publicized expert systems is American Express' Authorizer's Assistant which forms part of the company's credit authorization system. Authorizer's Assistant operates in real time and assists the human operator analyze a credit card transaction in view of authorizing it or identifying potential fraud. With this system, American Express can meet the projected growth for authorizations without significantly larger staffing requirements.

The Chase Lincoln Bank of Rochester NY has developed a Personal Financial Planning System which acts as an expert financial advisor to offer clients a comprehensive financial planning service. It includes a broad range of expertise including investment planning, debt planning, retirement savings, etc. The service costs clients $300 and compares to financial plans produced by traditional methods priced at $10,000. This expert system fills a strategic need for the bank to provide new services and does not require the hiring of trained personnel.

The Intelligent Banking System (IBS) was developed by a consulting company for Citibank. It is a family of applications developed to increase the productivity and effectiveness of English text message processing. IBS applies a combination of natural language processing and rule-based expert system techniques to analyze English language money transfer and other telex messages. The application was deemed to be a success if it could identify 80% of the information in an average telex and process it in under 30 seconds. IBS met these criteria and is considered to be cost effective.

MCI developed a similar system called the Automated Money Transfer Service which is a natural language understanding system that parses English language money transfer telex messages and converts them into formatted messages. With this system, the productivity of the human operator is increased as only the more complex or ambiguous messages need to be manually reviewed.

Mediatop, a French company, developed Television Intelligent Mediaplanning Expert System (TIMES) to help the mediaplanner prepare a television advertising plan. It can assist in the following tasks: definition of the TV campaign, construction of the campaign, audience forecast and preparation of a report. With TIMES, the design of a $5 million campaign can be achieved in minutes rather than days. Mediatop can react quickly to client changes and they see it as a significant competitive advantage.

Dupont has been a leader in the deployment of small expert systems and Packaging Advisor is a good example of their strategy. Packaging Advisor helps design rigid plastic food containers given general parameters about the use, shape and dimensions of the container. It analyses various design alternatives for material, thickness, barriers, cost, etc. and recommends an optimum solution. It is used as a sales aid for new resin products and the sales force estimates that 30% of their sales are attributable to Packaging Advisor.

Pacific Bell has developed the Enhanced System Monitor (ESM) which scans messages obtained directly from CPU consoles and alerts the operators of current or impending emergency situations. ESM can monitor up to 100 computers and it will announce priority alarms on a public address system using a speech synthesizer. In addition, ESM will also provide online diagnostic consulting services to the operator. ESM has helped reduce downtime of Pacific Bell's computers and it has allows the operators to be more productive. It is expected to pay for itself in less than three years.

Finally, I will mention the Ship Planning System (SPS) developed for the Port of Singapore. SPS is an intelligent planning tool to assist in the generation of schedules for container ship discharging and loading. Planners can prepare schedules in half the time with SPS which allows them to cope with the significant increase in container traffic.

GENERALIZATIONS FROM THESE EXAMPLES

You can probably determine from these examples that expert systems can and are being used in a diverse range of industries. They are used primarily as assistants to humans. Typically they provide assistance to the expert himself, handling the repetitive or less challenging cases and allowing the expert to spend his time more effectively on the difficult or obscure cases. Expert systems are also used frequently by lower level staff. They allow staff who are not experts to operate at a much higher level which lowers the training time, reduces the need for highly skilled staff and distributes knowledge across the company.

Many of these systems combined traditional programming languages and tools with the expert system tools to integrate with existing applications and data bases. Only rarely are expert systems developed as standalone applications.

We often see expert systems used in areas that have not previously been automated. These application areas may have resisted automation because of the very nature of expertise.

WHAT ARE THE ADVANTAGES OF USING EXPERT SYSTEMS?

Expert systems have particular value in application areas that are characterized by decision making based on skill, experience and expertise. Knowledge workers are those people who add value to an organization by making decisions using information and their expertise developed over years of experience.

With the availability of expert systems, we can now capture the experience, knowledge and heuristics (or rules-of-thumb) of experts within our organizations. We can retain their knowledge in an understandable and maintainable form so that we do not lose it when the experts retire or go on vacation. Moreover, we can now distribute their expertise to others in the organization and the experts themselves use the systems to improve their own productivity so they can work on the difficult problems.

Think about an expert system coupled with a computer performance tool so that the tool not only measures what is happening on your machine but interprets the information to recommend what action you should take just as the top experts in the field would.

Think about your own organization. Think of the knowledge bottlenecks in your organization where production slows and time is wasted because the knowledge necessary to advance the process is not available. Expert systems retain, consolidate and distribute expertise and knowledge. They provide intelligent assistance and have shown remarkable productivity improvements

HOW CAN EXPERT SYSTEMS BE INTEGRATED WITH INFORMATION SYSTEMS?

We have seen expert systems integrated in many different ways with existing information systems and there should be no surprise in that. Information technology currently plays a role in processing data and providing information to people so that they make make informed, intelligent decisions. Expert systems can provide the means by which we make information processing a pervasive technology.

First lets look at how expert systems can act as a front-end to an information system. Some good examples are quotation systems and configuration systems. In most organizations which configure an end product from its constituent parts, we find one or more human experts who translate customer requirements into a configured order. These people are experts because they must be certain that all the necessary parts have been accounted for, the parts work

together and the requirements have been met. Furthermore, they probably have to deal with special pricing configurations, inventory management issues, scalability of the end product, modifications required, new products being announced and a host of other issues. To do the best job to achieve the lowest price may require many iterations and a lot of work.

Many companies are using expert systems to help their sales people configure orders with the customer. The big advantage is that a prospect can get a price and a configuation while he is still interested in the product. The delay between a new product being introduced by the factory and a sales representative selling that product can be dramatically reduced. Moreover, the expert system has all the information necessary to feed an order processing system which probably reduces paperwork and duplication of effort.

What we are doing is distributing the knowledge so that the customers always interact with the sales representative and the expert system. If tradeoffs have to be made, the customer can make them.

A second important role that expert systems are playing is as a back-end to information systems. Expert systems are widely used in the financial sector to analyze data and provide recommendations which human experts would otherwise provide. Financial planning systems, taxation systems and systems that analyze financial statements are all in place and they examine hard numbers, the relationships between the numbers and the "soft" facts in the same way that human experts would to provide recommendations.

There are many financial planning expert systems in use today because financial institutions are now striving to provide more services to their customers. They also have a lot of very confusing products (or financial vehicles) that they would like to sell to their customers. It is simply too expensive to have a human expert do the analysis necessary to produce a financial plan. But we can capture the financial planner's rules and apply them in the same way to a client's financial position, goals and attitudes to produce a financial plan which will help the client make the decision of "is this a good investment or a bad investment for me?"

The idea of being able to package expertise or knowledge is a phenominal one. We simply cannot be experts ourselves in many areas. Expert systems allow institutions, organizations, companies and individuals to obtain packaged expertise to analyze their data quickly and consistenty. Maybe that is what the power of computing is really about. We see it happening in the financial area because of the dynamics of the industry. It is difficult enough for the experts themselves to keep up with the changes let alone the rest of us.

Finally, I will mention what we will call embedded expert systems. These are systems which replace or complement human expertise within a process so that the process will not have to stop or get bottlenecked. These systems include scheduling systems where the order has been placed in the system but we need help in expediting it. There is a lot of work and a lot of heuristics in determining an optimum schedule. Moreover, the schedule has to be recreated quickly because people call in sick and machines break down and customers demand rush orders and your tanker of crude oil arrived two days early from the Gulf.

Often these systems will provide one or more recommendations based on the rules and the current situation on how best to proceed and it may be completely hidden that there is an expert system functioning in the background at all.

WHAT IDENTIFIES A GOOD EXPERT SYSTEM CANDIDATE?

How do you know if your application is a good candidate for a knowledge system? The following checklist is a subset of the identifying features we use to help decide how good a fit the project will be for this technology. It is best if the domain is fairly limited and does not rely on common sense or sensory inputs.

- √ The domain is characterized by the use of expert knowledge, judgment and experience.
- √ The knowledge may be heuristic in nature and may require that decisions be based upon incomplete or uncertain information.
- √ The task is very clearly defined.
- √ Conventional (algorithmic) programming approaches to the task are not satisfactory.
- √ There are recognized experts who solve the problem today.
- √ The experts are better than the amateurs in performing the task.
- √ Expertise is not or will not be available on a reliable or continuing basis so there is a need to capture it because the expertise is scarce, expensive, dependent on overworked experts, or will be less available in the future.
- √ The system can be phased in and allows for some percentage of incorrect results.

√ Test cases are available.

THE NATURE OF EXPERTISE

We train people to have expertness but not expertise. We call people such as lawyers and engineers para-professionals - those who master sequences of activity. But they may lack the generative capability for creating those sequences that true experts have. In a knowledge-based system we have to be certain that we are getting the expertise underlying the performance.

Experts guess a lot. They are effective at estimating things in ways that less expert people cannot begin to comprehend. They work at hard problems, get things right most of the time and do that quickly.

But expertise is not just lots of experience. It involves the use of that experience to recognize key issues and ignore irrelevant ones. Human experts use macros that appear to be guesses. They cannot give IF/THEN rules or cause and effect rules directly. We expect them to reason intelligently rather than intuitively. The process tends to be invisible to the expert and they lack self knowledge. This is known as the Paradox of Expertise. They more knowlegdable a person becomes, they less able they are to explain their knowledge.

Human beings have developed the capacity to recognize and store away the things we do often to long term memory, like compiled programs, so we can run them in short term memory with far less expenditure of resources. The problem is that we no longer know what they contain.

Experts look for the shortest distance from problem to solution. Past experiences provide the shortest path. If not, novel problems can be a good technique to get at the underlying principles analytically using deductive steps.

Heuristics are used to avoid "garden path" results. Experts have meta-rules that say "be careful at this point".

Experts can gain a greater awareness of their knowledge by working with a Knowledge Engineer. Knowledge Engineering can create new knowledge by illuminating tasks where experts made mistakes or overlooked important things.

AN EXPERT SYSTEMS PROJECT

Small, prototype expert systems can be build with an expert system tool in three or four weeks. Moderately sized and extremely useful systems can be built by experienced knowledge engineers in six to ten weeks. These tools lend themselves to prototyping methodologies and they are meant to produce dynamic and easily maintainable applications very quickly. As a result, even longer term, more advanced expert system projects can be built in reasonable time-frames.

The steps that we go through in any expert systems project are as follows:

Application Assessment or Project Scoping:

Application Assessment is usually an initial meeting which lasts one day. During that time we often examine several applications and use an expanded version of the checklist provided above to identify the players, the commitment and how well the applications lend themselves to the technology. We can identify which, if any, of the identified applications appear to be good expert system candidates and we can help identify other potential applications as well. At the end of the day we come away with a feel for the scope of the projects, what interfaces will be required, an idea of the input and output and the the relative risk versus payback potential of the projects. From this initial scoping, the relative size of the projects can be roughly determined and usually we can present a proposal on how to proceed.

Domain Profile:

Domain Profiling typically takes two or three days. We like to have two knowledge engineers involved along with the project leader, experts and end users. Profiling can be an intensive session in which we map out all the facets of the domain and determine all the steps within each facet. Inputs, outputs, and interfaces are specified and the decision is made as to what, if anything, the expert system should do at each step.

Because expert systems are often implemented in areas which have not previously been automated, the profiling session can be the first time that the domain has been documented and it can be a very enlightening experience for all involved. It may be that management's view of how the process occurs is very different from that of the experts.

It is important to remember, however, that the purpose of our expert system is capture how the

expert is currently carrying out his function. We work from case studies to determine what decisions the expert actually made given a specific set of circumstances and determine the underlying rules. As in rules gathering, it is important not simply to have the expert speculate on how he makes his decisions. You must work from cases from an early stage or be able to postulate unique problems for the expert to solve.

The purpose of profiling is to determine the blocks of processing and control so cases and examples need only be concerned with the high level decision making. You should be careful to keep the session moving along. Do not get bogged down in too much detail and try to keep the focus within the bounds of the problem area.

By the end of the profiling session, we have determined what the deliverables will be over the full project. Often the project will have been scaled back or broken into several phases. The timing of each phase can be estimated quite accurately because often the initial rules are revealed and this becomes a decision point as to whether to proceed.

Rules Gathering:

Known as knowledge acquisition, the Rules Gathering phase often follows directly on from the profiling session. Whereas the profiling is an intensive session, rules gathering tends to be a very intensive session and usually lasts three to five days, depending on the number of experts and the size of the project.

Generally the same individuals will attend as were in the profiling session. The idea of multiple experts can become an issue. There will often only be one expert because you are trying to capture the expertise of your best person. In practice you may have two or more experts achieving equally good results but they may not use the same methods to get to their results. In that case combining their expertise may be of little value.

On the whole, we have achieved good results by using multiple experts. They can help clarify each other's reasoning. However, it is up to the project leader to resolve conflicts.

We base rules gathering as much as possible on case studies. These are important for a number of reasons. First, it documents the situations the system will have to cope with. Experts can be very misleading in trying to describe situations off the top of their head. Secondly, case studies show us the actual decision the expert made. If the rules we extract from the expert do not lead to that decision, then the rules are wrong or incomplete. Thirdly, case studies give us

acceptance criteria. Remember, the purpose of an expert system is to duplicate the expert's reasoning process over a limited domain.

Our purpose is to help the expert reveal his line of reasoning and we spend most of our time listening, clarifying and summarizing and keeping the process on track. Test cases are very important because we have found that experts usually attempt to give a very analytical explanation of their reasoning process which often bears no resemblance to the actual line of reasoning.

With test cases we can ask "What did you do first?" and the actual case can keep them focussed. It is very important to know the order in which they do things. What questions do they ask first? What initial guesses do they make? How do they recognize when they are pursuing a line of reasoning that is incorrect?

Without test cases the sessions can get bogged down and grind to a halt because the experts tend to concentrate on the extreme cases and exceptions to what they may perceive as the standard problem type. Cases bring the experts back to a focus which is concrete.

At the end of the rules gathering session, we should have all the rules, tables, formulae and interfaces defines. The user interface, the purpose of the software and the reports should be agreed to and defined as well. From here the knowledge is ready to be formalized for presentation to the users and approved.

Knowledge Formalization:

We write the rules manual during the Knowledge Formalization period. That may take up to two weeks, depending on the size and complexity of the project. The rules manual is our specification document. The rules represent the logic of the system and the rules manual shows the origin of the input, what the user is presented with, where the choices will lead and the outputs they produce.

The rules manual has to be done well. It has to explain each rule, table, and formula - what it means, why it is important as well as how and where it is used. Once written, we review the rules manual with the entire team to ensure that it is understood and complete. A software contract can be constructed at the same time as the rules manual. We identify milestones with their target dates and project the cost for the whole system or simply the first portion. We look for approval of the rules manual before proceeding but from here, the system is constructed to

implement the rules.

The rules can be formalized into a representation appropriate to the particular tool that will be used or they can remain in a more general form as obtained from the expert in the rules gathering session. It is important that the rules manual be readily understood by those who will be verifying it.

Prototyping:

Once the rules manual is complete and verified, the development can begin. We refer to it as Prototyping because the development process should be incremental and interactive. We schedule regular and achievable milestones which are usually delivery points as well. Those fall quite naturally from the domain profile.

Although we are in regular communication with our experts during this phase, we use the rule book as the specification. Due to the physical distance that frequently exists between our developers and the client site, we do not want to have an overly interactive development period where the expert changes his or her mind on a routine basis. We put a lot of effort into the domain profiling and rules gathering sessions so that we get agreement before we begin programming.

Test & Deliver:

In practice, the expert will be involved in the prototyping and testing phases to help refine the rules and prove them through the use of test cases supplied. One of the benefits of expert systems is the ability of the expert to test the system himself. With an explanation facility the expert can verify the rules and determine if any are not working correctly.

PROLOG

Our choice of tool in the development of expert systems is between the Prolog programming language and one of several expert system shells. Prolog's chief strength is that it is a general purpose programming language which gives us great flexibility in development, although it is a very high level tool.

Fifth generation languages like Prolog are valuable because they help reduce the distance between the verbal description of a process and its representation in executable code. The resulting programs are thus easier to both read and change. The logical foundations of a program tend to be closer to the surface in fifth generation languages than in the more conventional.

Computer languages have evolved from low-level languages in which the programmer specifies how something is to be done, toward high-level languages, in which the programmer simply specifies what is to be done. Most languages, Lisp included, are "how-to" languages. Prolog breaks from that, encouraging the programmer to describe situations and problems, not the detailed means by which the problems are to be solved.

Prolog is a programming language centered around a small set of basic mechanisms, including pattern matching, tree-based data structuring, and automatic backtracking. This small set constitutes a surprisingly powerful and flexible programming framework.

A Prolog program consists of a set of rules for deducing the truth of a given hypothesis (or goal) from a conjunction of other hypotheses. A set of rules with the same head represents alternative ways of establishing the same hypothesis. Such a set is sometimes called a procedure. Prolog procedures are somewhat analogous to procedures in mainstream languages and are conceptually like the Case statement. The goals in the body of a procedure are invoked sequentially and a procedure is exited when all hypotheses in one of its rules have succeeded.

However, a fundamental difference in the flow of control arises when one of the hypotheses fails. Whereas, in mainstream languages, failed statements must be handled explicitly, Prolog automatically backs up to the previous hypothesis, attempting to satisfy it in a different way. This behaviour is known as backtracking.

Programming in Prolog often consists of merely describing essential properties of a problem, defining the relations and rules applicable to the problem and stating known facts relevant to the problem. With conventional programming languages, the programmer has to spell out a detailed sequence of steps which the computer must perform. The declarative style of programming made possible by Prolog is faster, easier and less error-prone.

Prolog's symbolic nature, dynamic memory management and flexible structure make it an ideal language for the rapid prototyping of virtually any kind of system but Prolog is especially well suited for problems that involve objects - in particular, structured objects - and relations

between them.

The control structure in Prolog is unification or a pattern matching process operating on a sophisticated internal data base which contains a full relational data base. Data and programs are stored in this internal data base which is searched for solutions to goals. The Prolog system can search for alternative solutions (called non-determinism) by backtracking through the search space. Prolog programs in the data base consist of statements which express relationships between entities. The "logical engine" in the Prolog system then runs the program by inferring true statements from the given relationships.

Unlike conventional languages, Prolog incorporates program control into the language itself. The programmer is relieved of the majority of the control of program flow and can focus on expressing data objects' relationships.

There is currently no universal Prolog standard. Over time, the implementation described by W.F. Clocksin and C.S. Mellish in Programming in Prolog (Springer-Verlag, 1985) has emerged as the de facto standard. We use MPROLOG because it adheres to Clocksin and Mellish and it has source-compatible implementations available for the PC, 80386, HP3000, HP9000, workstations, DEC VAX, IBM mainframes and more.

EXPERT SYSTEM SHELLS

Expert system shells provide the knowledge engineer with a higher level tool to build expert systems more easily than with languages such as Prolog and Lisp. With such tools you are freed from the details of programming. Instead you can spend more time on designing your software, interviewing experts, and building effective user interfaces.

There are hundreds of shells available for all types of hardware platforms. Shells can range in price from under $50 to tens of thousands of dollars. Their capabilities and features are quite varied and many shells lend themselves to particular types of applications such as diagnostics.

It is important to bear in mind that expert system shells are still fairly immature and so most reasonably sized applications that have been developed with shells call external routines. Just as in the early days of fourth generation languages, you may find the PC shells restrictive. However, they generally have good interfacing capabilities and are reasonably priced.

The representation of knowledge in most shells is intuitive and quite straightforward. With

minimal instruction, the experts (whose knowledge is being incorporated into the software) can examine and validate the knowledge base. The main components of the system are objects to store variable data and specify where data is to be found; rules in IF-THEN format whose function is to incorporate the deductive logic of the expert system; formulas; tables; and expressions. Most shells offer both forward and backward chaining to suit the application requirements.

We find that it is best to chose a tool whose capabilities and features meet the requirements of the application, rather than looking for one shell that will have general applicability.

CONCLUSION

I hope that I have introduced you to idea of using expert systems within your organization. We have examined what expert systems are, how they are being successfully applied in commercial systems, how to identify if your application is a good candidate for an expert system and we have examined the steps involved in an expert systems project along with the tools available.

The use of this technology is of strategic importance in your organization. It can represent significant productivity improvements throughout your operation. Many companies are using expert systems to competitive advantage. For that reason, I suggest you investigate the technology. Identify a good candidate application and start small with a pilot project. From there you can evaluate the results versus the costs and decide what role expert systems can play in your future.

R. G. Hopmans
January, 1990

Automation, Not Aggravation: Software Development Techniques For the 1990's

Betsy Leight
OPERATIONS CONTROL SYSTEMS
560 San Antonio Road
Palo Alto, California 94306
(415) 493-4122

INTRODUCTION

Today's HP 3000 professionals are becoming increasingly concerned about software and data integrity. And as a result, implementing a reliable method of tightening control over the changes to systems and applications has become a prime objective of many shops.

An isolated change may initially seem harmless, but the ripple effect upon interrelated systems can often be devastating. As the volume of code used in a shop becomes greater, the effect of these change can become staggering. With an industry average cost of $50 per line, the software investment of even a small site can be valued well into the millions of dollars.

Despite the enormous value of today's software applications, much of it is vulnerable to inadvertent errors. The reason: MIS departments often use tedious manual procedures to track program changes, and these procedures are seldom if ever followed faithfully throughout the organization. Without an improved approach to monitoring how and why changes are made, a company's entire computer and software asset are at serious risk.

First, consider the case of our firm, Operations Control Systems. With over 4000 products installed throughout the world, our customer support staff receives requests for dozens of modifications every month. At the same time as these maintenance and enhancement requests are coming in, our R&D team is also under pressure to develop new products. Shipping a new release represents a major project for almost every department within the company. Without an efficient means to manage changes, maintenance work would forever delay progress on new development. Through the task of managing our own internal software maintenance and release management process, we at OCS developed much of our initial expertise in the area of change management.

Through feedback from our customers, we started to refine these techniques and make them available to the general HP user. When we first surveyed our customer base in 1986, we found that the application maintenance backlog was not unique to software developers. Some customers who relied entirely on vendor-supplied software with seemingly minor modifications ran as many as 6 months to 2 years behind schedule on their maintenance work. Thus, while many non-data-processing executives think that their MIS staff is focusing on new applications, the reality may be that MIS is bogged down in maintenance and change related activities.

This highlights a second major need for change management. Change management improves the ability of the MIS department to react to user demands for program changes.

THE NEED FOR CHANGE CONTROL

In order to understand the change control problem and how to solve it, let's explore the nature of the problem in an uncontrolled environment. The problem stems from the fact that in many HP 3000 shops, programmers can access production source directly. This can have several consequences.

First, consider an environment where multiple programmers make simultaneous changes to source code files. In this situation, the last programmer to update each file overwrites the changes made by other programmers. The result is frustration and wasted time synchronizing and retesting all of the changes.

Second, a careless programmer might inadvertently destroy the source code of a critical application. If a production error occurs and the source code cannot be restored easily, production could be delayed for hours or even days. The original author may not be available. The program may even have to be rewritten from an old copy, wasting development and test time and possibly reintroducing old bugs or creating new ones.

Third, the development procedures found in many shops eventually result in discrepancies between source and object code files. That is, the master (or "official") source files may not recompile into the current production object files. Since it is almost impossible to recreate source from object code, days might be spent searching for the correct version, recompiling it and validating it against the current production code. And, with all of this effort, there will be no guarantee that the source will exactly recreate the current production code.

These situations point out a fundamental rule of good change control: Programmers should never have unrestricted access to production files!

At first glance, these problems may appear obvious, but the fact is that many HP 3000 data centers continually react to the problems they cause, rather than introduce standards and controls to avoid them. After all, a data center is there to support its users. This charter forces management to focus resources on meeting daily user requests, distributing reports, managing hardware and completing batch production, rather than on implementing long-term solutions to problems that are not well understood. Although many shops can run error free for months, an innocent looking error can snowball into a major catastrophe.

If this is the orientation of your shop, you might want to consider this fact: the efficiency and accuracy of your work depends upon the integrity of your software assets. Is your data center a candidate for better change control? Try taking the following test.

What software systems do you have?	Where and why were the last changes made?
What programs comprise these systems?	Where are the appropriate backup copies?
How often do these programs change?	Which user requested the last set of changes?
Who is responsible for the changes?	What is the status of a specific change request?

If you cannot answer all these questions readily, your shop probably needs a better approach to managing changes.

Proper change control provides immediate answers to these critical questions. The result is more than just visibility and auditability. It is more than data integrity and improved maintenance efficiency. A good change control system ensures that every program change is authorized and controlled, making it possible to quickly restore systems when necessary. Change control techniques also make it possible to monitor changes made to software from the earliest stages of development through the testing phase and all the way through the move to production process.

Change control also improves the Quality Assurance process. It allows operations personnel to test new systems, moving software from development to test, or from test to production. And, all of this will be done with complete audit trails that allow the identification of each entity promoted and tested.

Another important aspect of change control is provision for management
inquiry in areas such as: the status of a change, the current version of a file, the change history of a file, and the impact of a proposed change. Supplying this information can often be a tremendous benefit to managing the software life cycle.

RECOMMENDED CONTROLS FOR ALL SHOPS

In order to achieve the benefits of change control, a shop should address each of the following five major categories:

Control through stages: Keeping control over the movement of software through the development and maintenance process.

Change history: Providing a complete history of every change requested and made to the source code.

Control over source and procedures: Maintaining control over users authorized to access source code and the processes that render source into executable form.

Security: Retaining control over the personnel who are authorized to make program changes.

Inventory: Maintaining an accurate inventory of all the programs, files and other components of each system.

A more detailed checklist identifying good change control practices appears at the end of this article.

With these items in mind, lets look at the costs and benefits of manual change control techniques.

MANUAL CHANGE CONTROL

HP 3000 users have developed a variety of creative procedures for controlling changes in their environments.

A common strategy requires programmers to FCOPY source code from the production location into a development location. This strategy can be implemented at three levels.

1. The development area may be nothing more than a set of individual groups where files reside. In this case, each programmer copies all of the necessary files into his own group where he makes the appropriate changes. There is no standardized testing environment.
2. Separate account structures may be maintained for production and development. Often a separate account structure is maintained for testing as well, to duplicate the production account structure. Thus, each programmer can be confident that testing is conducted in an environment that closely resembles production. Establishing separate account structures on a single computer often produces adequate results.
3. A separate development computer may be used, thus eliminating the possibility of direct access to master files located on the production computer. Although this approach is ideal, it is not possible without a separate development computer.

Unless stringent controls exist, however, this strategy does not guarantee that only one individual can check out a particular file at a time. And, since programmers are not restricted from accessing production accounts, there is no way to audit their access.

Once the development is completed, many shops allow the same programmer to test their own work and simply overlay the original production files with the new version of code. Since such a procedure seldom represents adequate control, it is widely suggested that a formal Quality Assurance (Q/A) process be used.

There are several ways to initiate such a process, depending upon the resources that are available. Smaller companies may have programmers Q/A test their colleagues' development efforts. In these cases, tests are usually performed in the development account.

Larger organizations may dedicate one or more individuals whose sole function is testing. Here, a separate Q/A test account is generally set up to reflect the production account structure. In this case it is vital that the developer is finished, all claims to the code are relinquished. After it has been moved to Q/A for testing, the one copy of the code will exist only in Q/A. If the Q/A analyst locates an error, the code will be returned to the original developer for revision. It is now Q/A's turn to relinquish all claim to the code by moving it back to the development location and purging it from Q/A. If simultaneous copies were to exist in both locations, last-minute changes in development might not be included in the Q/A version. Thus, the final production version would not be accurate. It is surprising how often this obvious safeguard is overlooked.

At the conclusion of the Q/A phase, another step will often exist. A higher level manager will perform a final approval on the development code to certify that all standards have been met and all tests have been satisfactorily completed.

Following final approval, the finished code is ready to be moved into the production location. Since the new files will be copied into production with the same names as the original files, the original files must first be backed up to tape to avoid loss of the earlier version. Next code must be recompiled to assure an exact match between source and object. Lastly, JCL and other files must be updated.

This update process is tedious, time-consuming and error prone, especially when large numbers of files are involved. Nevertheless, precise standards cannot be eliminated at this final stage or the shop will run the risk of serious inconsistencies.

Based on the previous discussion, it is possible to state several general rules for good change control:

First, establish separate accounts for production (master files), development and, if possible, test. Test accounts should be mirror copies of production. This allows files to retain their original FILE.GROUP names as they are moved from test to production and makes it possible to visualize the link between a developing program and its production/master version.

Second, when files are "checked out" from the production location to development, a copy should be made. The original source should never be destroyed. However, when development is complete, files should be moved to the test location, thus eliminating synchronization problems with old versions.

Third, when Q/A has approved all changes,, a project leader or manager should verify that adequate and accurate test procedures have been followed. Only at this point should code be moved into the production library. If system loads are heavy, such updates could occur in batch and should be initiated by operations or a production Librarian. The timing of the release into production should be coordinated with the user's schedules, with the release of other modules and with documentation changes, etc.

Prior to updating the library, the original production copy should be verified and stored. Without this step it is much more difficult to return to a prior version in the case of a problem.

As a general rule, we recommend that the authority to release files to production be restricted.

AUTOMATED CHANGE CONTROL

Although this article has described the minimum requirements necessary to achieve reliable change control, implementing these controls places a considerable burden on everyone involved in the change process . . . unless the process is automated.

Although an automated change control system will require some initial planning, once it is operational, the burden on the MIS staff will actually decrease. The process would be as follows:

The first step in implementing an automated change control system is to define your development environment. Decide what files you want to control, what types of file movements will be performed, what approvals will be required at which stage, and who will be authorized to perform or authorize each type of file movement. This will depend on such things as:

The size of your development staff
The levels of management and job responsibilities in your development group
The size, complexity, and volatility of your applications
The amount of packaged software used by your firm
The physical hardware configuration of your computers
The degree of control which you and your auditors agree is appropriate.

The following examples show four typical development environments, presented in order of increasing complexity.

As you read through them however, keep in mind that these are only intended to be representative examples; each shop is different. A good automated change control system must be configurable to meet the specific needs of the organization.

EXAMPLE 1 - BASIC DEVELOPMENT ENVIRONMENT

The basic development environment is a small shop with a single computer and a staff consisting of two programmers, one day-shift operator, and a "shirtsleeves" manager. The development control objectives are basic, but critical:

* They need to know where the current production versions of all source, object, and job files are,
* They need to be sure that all changes are made to copies of those files in a separate "test" location,

* They need to assure that all changes are approved by the manager before they are put into production, and
* They need to do this without further burdening their staff who is already working long hours.

The first step here, as for most installations, is to identify all of the production source, object, and job files. These files need to be grouped together and secured. Through an automated change control system, this can be accomplished by creating customized filesets which represent whole groups of files. This eliminates the need to change the operating system account/group structure or move any of the files around.

The second step is to define the file movement policy, or steps, that will be allowed in the shop. In this example, we have three basic file movement steps:

1. A CHECKOUT procedure, which copies source, object, or job files from the master location into a development location,
2. An APPROVE step, which allows the manager to stamp a changed file with their approval, and
3. A CHECKIN step, which allows the programmer to move a group of files back into the master library when they have been approved. This step also makes an automatic backup copy of the old master file. In addition, the backup copy is compressed to a small fraction of its original size to save disc space.

This process is illustrated below:

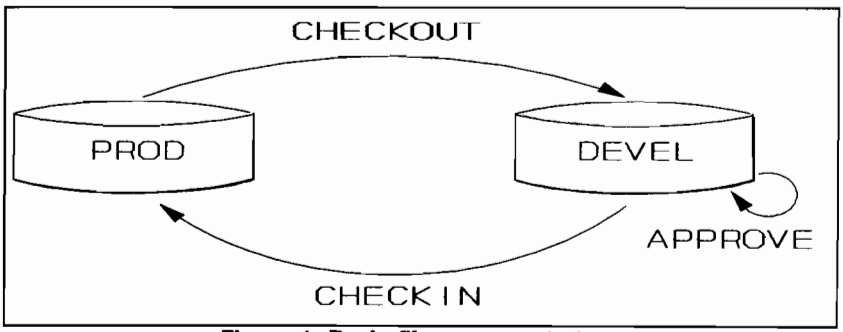


Figure 1. Basic file movement steps

To maintain the security of the master library using MPE alone, the CHECKOUT and CHECKIN steps would have to be performed by a manager, an operator or an additional employee. The approval step would have to be documented on paper. With an automated change control system, the CHECKOUT step can be defined to make test copies of the master files even though the master library is secured. The automated system can keep track of these copies and can prevent multiple programmers from checking out copies of the same file at the same time. The APPROVE step can be defined to mark the file(s) as approved, and can be restricted so that only the manager can use it. Furthermore, the CHECKIN step can be defined to allow the programmer to push his changed, approved files from his test environment into the master library without having to log on to the master account, and automatically archive the old versions.

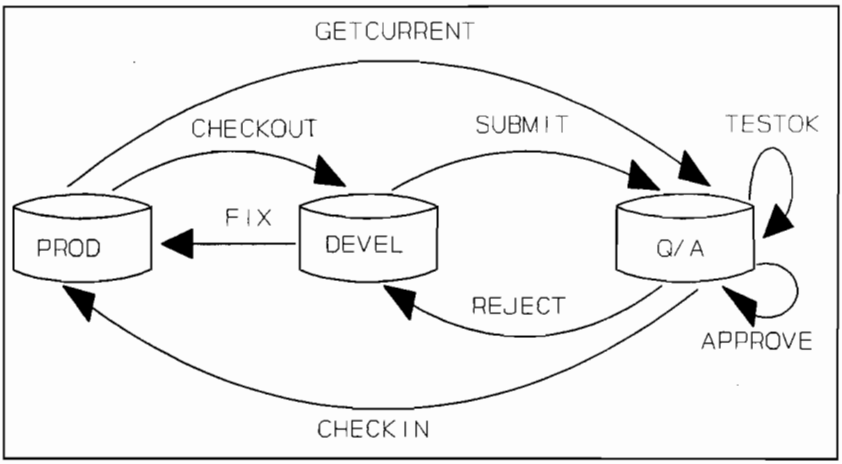
EXAMPLE 2 - SEPARATE Q/A FUNCTION

This example illustrates a medium-sized shop with a tightly controlled development process. The MIS organization consists of a manager, six programmers, two operators, and a full-time quality assurance staff of two. Due to the critical nature of their applications, this shop insists that their Q/A staff perform full unit and system testing on every change. They also require management approval of the testing procedure before any new or changed software can be put into production. They do, however, need to allow their programming staff to make "quick fixes" on an emergency basis, thus bypassing the delays that would normally accompany the Q/A process. It is especially critical that a complete and reliable audit trail be maintained for these quick fixes.

We can define this environment as follows:

1. A CHECKOUT step as described in Example 1. This allows the programmers to easily make development copies of the secured master files, and assures that only one copy at a time is modified (other concurrent copies can be made with read-only access).
2. A SUBMIT step, to allow programmers to move their modified source, object and job streams into the Q/A area.
3. A GETCURRENT step to bring read-only copies from the master library directly into the Q/A area. This allows the Q/A staff to perform an integrated test using the current production versions of programs that are not currently being changed.
4. Since not all changes will pass Q/A, we need a way to get the files back into the development location. We define a REJECT step which will be performed by the Q/A staff when a program fails testing.
5. Q/A will signify that a change has passed its system tests with a new step called TESTOK. This will help the manager keep track of the status of work on various programs and is a positive indicator that testing is complete.
6. We still need an APPROVE step, but it is defined to operate on files in the Q/A location. The APPROVE step will be performed only after they have been TESTOK'd.
7. The CHECKIN step now moves files directly from Q/A to the master library, once they have been APPROVEd.
8. Finally, we define a FIX step, which moves files directly from the development area to the master library.

This development environment is illustrated below:



Here again, the automated system facilitates file movement, emergency fixes, and management approval without burdening their staff. The CHECKOUT, SUBMIT, REJECT, and CHECKIN steps are defined to operate on specific groups of files, so users do not need to fully qualify file references. Wild-card references may also be used to move groups of files at once. These steps can be defined to automatically purge the files from their original location after copying to eliminate any need for additional housekeeping efforts.

Because an audit trail is maintained for each step, no extra effort is required to control use of the emergency FIX capability - the manager gains complete visibility by simply listing all uses of the FIX step.

EXAMPLE 3 - NETWORKING, SOFTWARE DISTRIBUTION, SOURCE/OBJECT SYNCHRONIZATION

The primary difference between this example and the two previous ones is that there are separate computers for production and development. In addition, the company's applications run on multiple remote computers. As in previous examples, the master library contains the production source, object, and job files, but in this case, the object and job files are actually executed on the various remote computers. This adds the important control objective of assuring that all of the remote sites are running the correct versions of the code. To this we will add the audit requirement of

source-object synchronization: we must assure that the object files in the master library (and on the remote computers) were in fact generated from the source files in the master library.

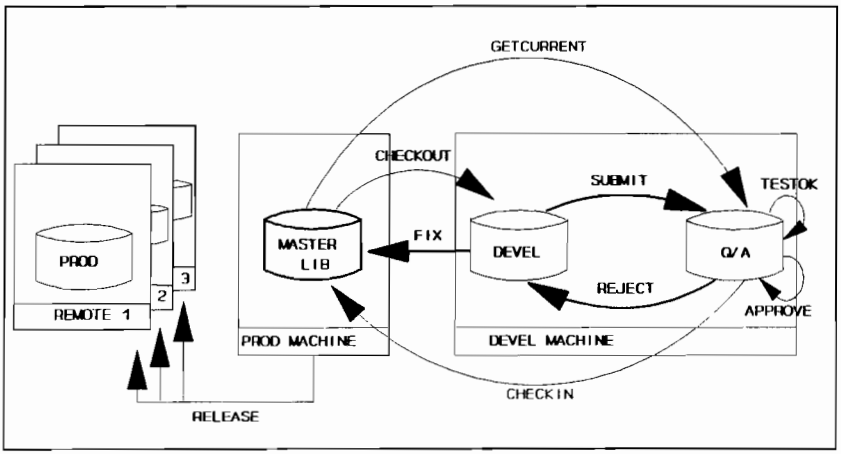
The rules and file movement steps for this environment are the same as for Example #2, with the following exceptions:

1. A new step, called RELEASE, is defined to distribute groups of object and job files to the remote computers. This step will copy the files specified to all of the remote computers, and produce an audit trail to verify that the copies were successfully transferred.

2. The CHECKIN and FIX steps will be modified to move only source and job files into the master library, and to automatically stream compile jobs to generate the object files from the new source. Since the new source has already been compiled to test the changes, this step is redundant, but it is an effective way of assuring that the source and object files always match and are synchronized.

3. While the remainder of the steps function as they did in Example #2, the automated system makes the multi-computer environment transparent to the development staff. Now CHECKOUT, GETCURRENT, CHECKIN, AND FIX all operate between the production and development computers without requiring extra steps or commands.

The resulting environment is illustrated below:



EXAMPLE 4 - PACKAGED SOFTWARE, ADVANCED VERSION CONTROL

In this example, the company uses third-party application software and receives periodic releases of the software from the vendor. Their own programmers have also customized portions of the software in-house. They have developed custom reporting and other extensions to the software. Whenever this shop receives a new release from the vendor, the new software is placed into its own separate account. There it is integrated with existing software by the programming staff, tested by the Q/A group and eventually put into production in the same manner as internally developed software.

Since the software is run on a large number of remote computers, the company releases newly approved production software in stages. First, a small number of remote users conduct a "beta" test for a limited period. When this stage is satisfactorily completed, the software is released to the remainder of the sites. Because of this procedure, the company needs to maintain two or more versions of the software simultaneously, and, in emergency situations, make changes to a prior version without interfering with work performed on the new version.

The rules and file movement steps defined for this environment are the same as Example #3 except for the following:

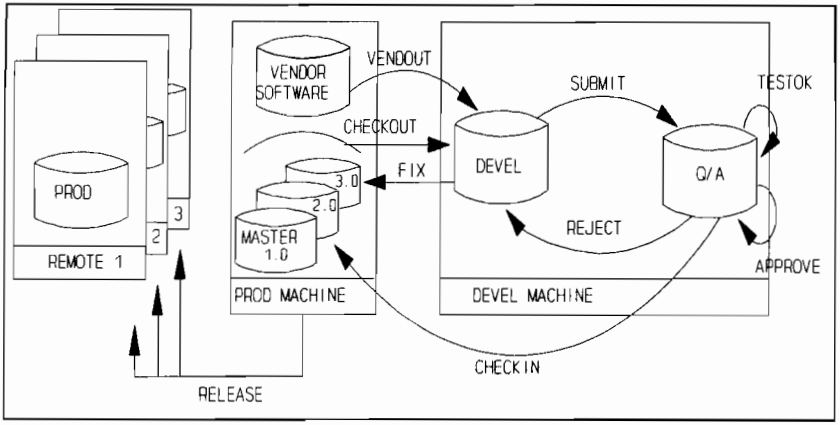
1. A new step, VENDOUT, has been defined for programmers to check out source and job files directly from the vendor software location. The step will prevent the programmer from accidentally getting an old version from a different location.

2. The CHECKOUT step is redefined to select the latest version of the software from the master library. This is accomplished by searching the "new release" location first. If the file is not found there, the "old release" location is searched. This assures that the programmers do not inadvertently use old versions of the software.

3. The CHECKIN and FIX steps are redefined to move files into the "new release" location.

4. The RELEASE step now operates on the version locations much as CHECKOUT does. When specific files are released, the automated system searches for the most recent version, and finds it even if it has not been changed for several versions. If a "general release is made, only the files that have changed are distributed. This reduces tape and disc storage requirements as well as network distribution costs by eliminating unnecessary file transfers

The resulting environment is illustrated below:



It is important to note that this rather complicated development environment can be precisely controlled with minimal effort through the careful choice of predefined file movement steps. Searching through two or more locations for the most current version can be performed automatically by the automated change control system. This enables the shop to maintain the integrity of prior versions while building a new version . . . without replicating files that have not changed. It also assures that programmers always get the most current version of any file.

SUMMARY

HP 3000 data centers have built a substantial asset in their software and data files. Yet, without effective change control, it is virtually impossible to guarantee the integrity of modifications to this valuable asset.

Furthermore, industry surveys show that most firms have a substantial software maintenance backlog. Without efficient means to manage changes, this backlog severely hampers the progress of new development efforts.

As a result, it is becoming quite common for HP 3000 shops to establish formal procedures for software change control.

Although many shops initially attempt to implement change control through a manual system, these systems have met with limited success due to the heavy burden they place on the MIS staff and the

inherent unreliability of the manual approach. Because automated change control systems provide an effective, reliable solution while reducing the burden on the MIS staff, they have recently become the standard. These automated systems improve development and maintenance efficiency and ensure that all program changes are properly authorized, documented and tracked. Automated systems provide management with quick access to the status of changes, the change history of a file and the impact of a proposed change. They provide programmers with an environment that allows them to concentrate on programming rather than on searching for the correct versions of files and documenting their usage.

Safeguarding the integrity of the software asset while paving the way for more efficient productive development and maintenance, is one of the most compelling challenges facing HP 3000 professionals today. Automated change control systems can provide the tools to meet this challenge.

CHANGE CONTROL CHECKLIST

Change control procedures for computer programs should be established and followed. The intent of these controls is to prevent unauthorized, inaccurate, and unreliable program changes from being incorporated into the live production environment. Both scheduled and emergency changes must be appropriately controlled to maintain the ongoing integrity of software.

You can use the following techniques to ensure that proper controls are being maintained over your program changes:

- Develop and adhere to formally approved written standards for all program changes
- Define and enforce procedures detailing who can initiate and who can authorize program change requests
- Describe and track the nature and reasons for proposed changes
- Enforce testing and acceptance procedures for all program changes including emergency changes
- Test all program changes under normal operating conditions
- Involve users in preparing test data and reviewing test results
- Investigate and correct all errors before transferring code to production
- Certify that all test results demonstrate adequate protection from fraud, waste, and misuse of the program
- Document all program changes and update appropriate documentation as changes are made
- Log all completed changes as well as those changes in progress
- Utilize a formal system to report all changes to users and project managers
- Enforce a checkout-checkin procedure that prevents a file from being simultaneously modified by more than one programmer
- Develop procedures to analyze whether other systems are affected by new program modification
- Retain and secure original source code until changes have been processed, tested and updated
- Limit the frequency of program changes, except for emergency cases
- Notify both the user and EDP project manager when emergency changes are made

SOFTWARE DEVELOPMENT - METHOD OR MADNESS

3008

Debrah Whitaker
Albuquerque Publishing Company
7777 Jefferson N.E.
Albuquerque, New Mexico 87109
505-823-3240

Introduction

The primary purpose of this paper is to promote the use of methodologies (any methodology) in analyzing, designing and developing software. It is the belief of the author that although techniques and tools have been widely available for over ten years, most software is developed by programmers who are not taking the time to define, analyze, document or structure the system in any form before the coding process begins. Additionally, it is the premise of this paper that because HP systems are more "programmer friendly" than other systems, HP users are less likely than other hardware communities members to focus on utilizing a formal methodology.

It is important to note that it is not the intention of this paper to sell, promote or endorse the use of any single methodology or product over any others. Various products will be mentioned throughout this paper with the intention of providing the reader with some frame of reference for available tools and workable solutions.

Why is using a methodology so important? James Martin and Carma McClure make the important point that "programming is not yet a precise discipline because the discipline itself is fairly young and full of brilliant people who want to make up their own rules." These authors further state, "one of the primary reasons that building and maintaining software systems is so expensive and error-prone is the difficulty we have communicating our ideas to one another... the larger the team, the greater the chance for error." (1)

Square One

It is important to understand some basic definitions prior to discussing the available products. Some or all of these terms may be familiar to the reader, and if so, the author apologizes and suggests that the reader skip directly to the next major section of this paper.

A key concept which is the basis for understanding the nature of software development is the System Development Life Cycle or SDLC. The SDLC is based on the concept that system development can be:

- broken into discrete phases or stages
- provide interim end products within specific time frames
- allow for review and testing throughout all stages
- provide for clear and concise task lists

1. Martin, James and McClure, Carma. Diagramming Techniques for Analysts and Programmers. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1985, Page 3.

Typically, the Systems Development Life Cycle is broken down into seven phases or stages:

- feasibility study
- user requirements definition
- system definition or specification
- system design
- program development and coding
- system test
- implementation and production

It is important to note that the correct use of the SDLC is not linear. The stages of SDLC should overlap and, in fact, be iterative. The SDLC is based on the premise that the key to proper systems development is in performing the "right" work up front. Research has shown that for every one week spent on the requirements phase, project teams can save one month on the program development phase.

The word methodology, as defined by Webster, is "1: a body of methods and rules followed in a science of discipline 2: the study of principles or procedures of inquiry in a particular field." When discussing systems development, methodology typically refers to a specific tool or technique such as Data Flow Diagrams, Warnier-Orr Diagrams or Structure Charts. Each one of these tools or techniques is based on an overall approach and philosophy relating to system development.

CASE (Computer Aided System Engineering) is a more recent concept referring to interactive diagramming, programming, and documentation techniques. The underlying philosophy suggests automating the entire system development process including validity checking, consistency checking, creating and editing of calculations, and ultimately code generation and program documentation. In summary, CASE may be thought of as the automation of a methodology.

Problems, Issues

As anyone who has lived through the SDLC of a system knows, anything can go wrong. As Tom DeMarco so aptly stated in 1978, "Projects can go wrong at many different points. The fact that we spend so much time, energy, and money on maintenance is an indication of our failures as designers; the fact that we spend so much on debugging is an indictment of our module design and coding and testing methods....when analysis goes wrong, we don't just spend more money to come up with a desired result - we spend much more money, and often don't come up with any result."(1)

1. DeMarco, Tom, Structured Analysis and System Specification. Englewood Cliffs, N.J.: Prentice-Hall, Inc. 1978, page 9.

It appears that we are still waging the same battle today that DeMarco described 12 years ago. More recently, James Martin and Carma McClure summed up the issue by saying, "One of the problems with computing is that it is so easy to make a mess..., most specs for large systems are full of ambiguities, inconsistencies, and omissions." (1)

Research shows that systems fail for a variety of reasons including:

-Resource Failure: people and time are insufficient to build the required system
-Requirement Failure: incomplete, incorrect, unclear specifications
-Goal Failure: inadequate or incorrect statement of goals
-Technique Failure: lack of use or incorrect use of proven software development disciplines
-User Contact Failure: inability to communicate with the user community
-Organizational Failure: inability of the organizational structure to support the system building process
-Technology Failure: failure of acquired hardware/software utilized by the system (2)

In summary, systems fail due to premature installation of hardware or software, lack of user involvement in all phases of SDLC, inconsistent design practices, absence of documentation and/or lack of training. It is important to note that system defects can, and will, be introduced at all stages of the SDLC. As Dr. Judith Schlesinger says "..bugs are cute--defects are not".(3)

Solutions

Numerous articles appear on our desks on a daily basis describing projects that fail and reporting the cost associated with such a project. Very little coverage is given to projects that succeed. However, research has shown that successful projects have a common set of characteristics:

-a clear vision of the desired process and the desired outcome, or goal, was developed

1. Martin, James and McClure, Carma. Diagramming Techniques for Analysts and Programmers. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1985, Page 3.

2. Schlesinger, Judith D. Structured Methodologies. Denver: Management Development Foundation Handbook, 1989.

3. Ibid.

early on in the project life cycle.
-a prioritized list of actions was developed along with a plan to accomplish these actions; the plan was executed and progress was tracked against the plan.
-project leaders understood the status of the process throughout the project.
-the required resources were committed.
-the goals were updated throughout the project as required.
-a structured methodology was used.

If used correctly, a proven methodology should address all or most of the issues before the project became a failure. Most of the structured methodologies utilize some form of structured resource and task planning; a clear cut form for documenting and diagramming goals and requirements; and these methodologies advocate the use of proven structured development discipline. Additionally, almost all of the methodologies are highly dependent on user involvement throughout the Systems Development Life Cycle.

Although organization failures and technology failures are not directly addressed in many of the available methodologies, each philosophy, if properly applied, will allow for discovery of these failures early in the development process.

Available Methodologies

Certainly more methodologies exist then can be reviewed or discussed within the context of this paper. The methodologies chosen for this paper are ones with which the author is the most familiar. The reader should only consider these as a small, but representative, sample of methodologies which have been successfully used on both large and small development projects.

Methodologies can be grouped, for the most part into three major categories or approaches; functional structured analysis and design, object-oriented structured analysis and design and technique-oriented analysis and design. Additionally, each methodology has strengths and weaknesses associated with the approach. The following section of this paper will attempt to identify the category or approach associated with each methodology and will further point out strengths and weaknesses with each approach.

Data Flow Diagrams - Data Flow Diagrams (DFD's) are actually a subset of an approach called "Structured Analysis" originally envision by Larry Constantine and later refined by Tom DeMarco while working for Yourdon,

Inc. The DeMarco approach is over ten years old and has become the standard in many large MIS shops throughout the U.S. and in Europe. DFD's are a top-down, non-narritive (pictorial) approach to developing functional specifications. DFD's identify (from the view point of the data) the sub-systems and each of the related interfaces. DFD's offer a representation of the physical model and of the logical model of a system as well as identifying the relationship between the physical model and the logical model.

Data Flow Diagrams are excellent tools for describing, communicating and documenting data, interfaces, major functions and the relationships among these functions. Users find Data Flow Diagrams easier to understand and use than most other tools. Data Flow Diagrams, in isolation, do not show programming logic or programming constructs very well. (See appendices A-1 and A-2 for a sample Data Flow Diagram and for the specific constructs used in DFD's.) In the context of structured analysis, high level data flows are "exploded" into lower level data flows. At the lowest level, DFD's are used to produce Structured English charts and Data Dictionaries.

Action Diagrams- Action Diagrams are a method of diagramming and documenting high-level decomposition, overview program structures, and detailed program control structures. Action Diagrams are a derivation of structure charts and Warnier-Orr Diagrams that were refined to accommodate all levels of system development. The primary strength of Action Diagrams is the "cradle to grave" usability of a single methodology. (1) This approach ensures more consistency across all phases of the Systems Development Life Cycle. Action Diagrams work well with fourth-generation languages and can be tailored to specific "dialects." The primary weakness of Action Flow Diagrams is that they are more difficult for unsophisticated users to understand than other methodologies such as Data Flow Diagrams. (See appendix B-1 and B-2 for a sample Action Flow Diagram and for a description of the specific constructs used in this methodology.)

Action Diagrams use a combination of narrative and non-narrative constructs. Brackets ([) are the building blocks of Action Diagrams. The way in which the brackets are drawn indicates repetition, exclusivity, conditions, or loops. Other constructs are used to indicate subprocedures.

1. Action Diagrammer, from DDI, Ann Arbor Michigan

Warnier-Orr Diagrams- Warnier-Orr Diagrams are a form of functional Structured Analysis and Design named after creator Jean-Dominique Warnier and refiner Ken Orr. Like Data Flow Diagrams, Warnier-Orr Diagrams have been available and highly used throughout the U.S. for a number of years to aid in the development of structured programs. This methodology was really the first formal structured design tool utilized in this country. Warnier-Orr diagrams are popular because they are easy to learn and use; they are composed of only four basic constructs or techniques. Warnier-Orr diagrams present the hierarchy of a process, program or a system graphically. The presentation is drawn horizontally across a page using brackets rather than vertically down a page using boxes or circles. Each layer of brackets represents a functional decomposition of the item at the left of the bracket. (See appendices C-1 and C-2 for a sample of Warnier-Orr Diagrams and more description of the constructs used.)

Structured English -Structured English is a narrative method of expressing design specifications. This was originally used to replace flowcharts as a more thorough method of depicting program detail. Structured English simply uses the English language combined with some basic structured programming techniques to depict a module or process. The real strengths of Structured English are that 1) it is very easy for most programmers (and maybe sophisticated users) to understand, and 2) it is fairly easy to translate into structured code. The primary weakness, as Martin and McClure point out, is "even at the level of structured English,... a picture is worth a thousand words. Analysts and users make mistakes with structured English because it does not reveal the logic structure with same clarity as a good diagram." (1) (A sample of the structured English constructs is shown in appendices D-1 and D-2.)

Even with its weakness, Structured English is an excellent methodology to use in conjunction with one of the non-narrative techniques such as Data Flow Diagrams or Nassi-Shneiderman Charts.

Nassi-Shneiderman Charts- A Nassi-Shneiderman Chart is a technique primarily used for detail program design. This technique was developed by I. Nassi and B. Shneiderman to replace traditional flowchart deficiencies with a structured view of programming

1. Martin, James and McClure, Carma. Diagramming Techniques for Analysts and Programmers. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1985, Page 165.

logic. (1) Nassi-Shneiderman (N-S) Charts are an excellent way to show detailed program structure and design. N-S Charts are an excellent way to force designers and programmers to use structured techniques. If the detailed logic cannot be shown on an N-S Chart, the logic is probably not structured. (See appendices E-1 and E-2 for a sample N-S chart.) Nassi-Shneiderman charts are not good at visually showing the relationship among program modules or at indicating hierarchical control structure. (2)

Nassi-Shneiderman Charts are a very effective tool when used with some other methodologies, such as Data Flow Diagrams or Warnier-Orr Diagrams. This technique was clearly designed to be used effectively for detailed design and can be used to strengthen other tools which are more effective in earlier phases of SDLC.

Prototyping/Iterative Development- Although the concepts of prototyping and iterative development have been around for a number of years, only recently has an entire philosophy of applying these concepts been formalized. In particular Stephen M. McMenamin in conjunction with The Atlantic Systems Guild, Inc., has developed a whole methodology based on iterative development. Mr. McMenamin claims that paper models cost too much time and money and in fact are incorrect a good deal of the time because they can never precisely reflect reality. Iterative Development allows designer/programmers to jump into the programming stage early in the project. Iterative development also allows users to get the real "feel" of the system much earlier in the development process. This approach allows programmers to "throw away" one or more generations of the product before development is complete. (Author's note: since prototyping is an automated process, no sample appears in the appendix.)

The primary problem with Mr. McMenamin's approach is that the resources required to use this method would be extremely expensive and most MIS shops cannot afford to "throw away" the first generation of software. In all fairness to Mr. McMenamin, the cost of CPU, disk, and memory are all falling so this approach may become more feasible with time.

1. Nassi, I. and Shneiderman, B. "Flowcharting Techniques for Structured Programming, " ACM SIGPLAN Notices, Vol.8, No. 8 (August 1973): pp 12-26.

2. Martin, James and McClure, Carma. Diagramming Techniques for Analysts and Programmers. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1985, Page 181.

Automated Tools For The HP Environment

The reader may ask the question, "Why bother using an automated or CASE tool?" Martin and McClure effectively answer this question; "Old techniques are obsolete. Most (diagramming) templates still have symbols for magnetic drum, punched tape, etc. Today, we use structured techniques but these techniques need to be automated..., The true structured revolution is that which makes the programmer unnecessary by the use of code generators working from structured designs." (1)

CASE purists argue that a tool cannot be considered "CASE" unless the tool is totally automated and unless the tool addresses all phases of the Systems Development Life Cycle. If we were to use this definition, the author would have to admit that no "real" case tools exist in the HP environment. However, most software developers agree that automated tools exist which can be effectively used to assist designers and programmers in one or more phases of the SDLC. Unfortunately for the HP community, most of these tools have been developed for IBM and DEC based systems. Maybe someday, all software, including CASE tools will have a common platform and the hardware will be transparent to the developer. Such is not the case today. The issue of hardware-dependent and operating system-dependent design remains a consideration The remainder of this section will be dedicated to reviewing products that can be classified under a more liberal definition of CASE and can be utilized for development of software on HP systems.

The products available on the market today vary in cost and in sophistication and range from basic PC-based drawing tools to HP-3000-based development tools.

Drawing Tools

Drawing tools are PC-based tools that can be used as automated templates. Typically, drawing tools allow the user to place different shapes on the screen and to move and label these shapes according to the desired outcome. These drawings can be stored on disc and printed for presentation or for later use in documentation.

HP Gallery Collection- The HP Gallery Collection consists of two products: HP Drawing Gallery and HP Charting Gallery. Charting Gallery is designed to create sophisticated, presentation-quality pie/bar/line charts bases on spreadsheet information. Drawing Gallery,

1. Martin, James and McClure, Carma. Diagramming Techniques for Analysts and Programmers. Englewood Cliffs, NJ: Prentice-Hall, Inc. 1985, Page 4.

however, may be used as an automated template to create diagrams in support of the non-narrative methodologies described earlier in this paper. Drawing Gallery is sold with a library of various pictures, shapes, lines and arrows. This software works best on a Vectra 286 with an EGA or VGA monitor. The Gallery series comes with a PaintJet driver so that drawings or charts may be printed in color.

MACDRAW- The MACDRAW product, which only works on Macintosh/Apple products, can be a useful alternative to the HP Gallery products. Like Gallery, MACDRAW is sold with a library of pictures, shapes, lines and arrows which can be utilized to produce drawings based on various methodologies described earlier and, like Gallery, this product only automates the drawing task.

Even under the most liberal definition, HP Gallery and MACDRAW can hardly be considered CASE products However, if other tools are unavailable, these products at least offer a method to quickly make design changes without redrawing everything.

PC Based Tools Which Support Methodologies

Automated PC-based tools have been developed to support most, if not all, of the non-narrative methodologies. These tools vary in price, functionality and ease of use. Most of all they vary in the level and manner they are connected with the various phases of the Systems Development Life Cycle.

Analyst Tool Kit- Analyst Tool Kit (or ATK) is a product that was developed by the Chinook Software Company of Denver, Colorado. ATK was originally developed to assist Instructors in teaching Data Flow Concepts and general design concepts to students. The ATK product contains a data dictionary and a library graphic shapes (circles, boxes, curved arrows) which are typically associated with the Data Flow Diagramming techniques proposed by Tom DeMarco. ATK will perform consistency editing but will allow the designer to override any error messages and proceed with a design.

Excelerator- Excelerator is a PC-based product that was developed by a company called Intech of Cambridge Massachusetts. Excelerator is a tool which supports a variety of non-narritive methodologies such as Data Flow Diagrams. This product provides links between the diagrams and the data dictionary.

METHOD/1- METHOD/1 is a combination of manuals and PC-based tools developed by Arthur Andersen Consulting, formerly Arthur Anderson & Company. METHOD/1 is a

cookbook approach to systems development that can be very effective when dealing with a very large project requiring multiple design teams or when dealing with new analysts with little or no prior design experience. The primary weakness of METHOD/1 is it's potential for burying the design team in piles of paperwork. METHOD/1 was originally designed to be used in support of Warnier-Orr diagrams however the product can be customized to be utilized with other techniques and approaches.

ATK, Excelerator and METHOD/1 are effective tools for documenting the analysis and design phases of the Systems Development Life Cycle. These tools save time and money through the reduction of manual effort spent in making changes to documentation during the systems development process. All of these products contribute significantly to reducing errors caused by inconsistencies among diagrams or among designers.

Tools that Support Detailed Design and Development- In the past year, several products have emerged which have the potential of becoming full CASE products. These products currently support the "back end" of the SDLC - detailed design and program development.

Speedware- Speedware is a mini/ mainframe product that was developed by Infocentre Corporation of Ontario, Canada (offices throughout Canada, Europe, and the U.S.). Speedware was originally designed as a 4GL language called "Reactor" and now is sold as a development environment. The product comes with:

Reactor - a 4GL language
Designer - an interface with Reactor that supports prototyping
Documentor - a tool that creates documentation after development is complete

A PC development version of Speedware is also available.

Softbench- The Softbench product is a product developed by HP initially for its scientific and engineering customers. The product was announced in July of 1989, it is expected that HP will eventually focus on the business customers. The Softbench product will create a common development platform for multi-vendor environments using a Windows format. Users of Softbench will be able to write code on one CPU and compile the code on one or more CPUs using Appolo's Network Computing System (NCS).

It is important to note that the author has seen presentations of both Speedware and Softbench and has discussed the products' features with vendors,

however she has had no "hands on" experience with either of the products reviewed. Both of these products look promising for the HP user community in terms of CASE technology.

CASE Technology Tools- Unfortunately, at the time this paper is being written, there is nothing on the market that can be considered a total CASE technology tool for the HP environment. Such tools are being marketed for IBM, DEC and other systems. There is little doubt that sometime in the not too distant future, CASE technology tools will make current techniques obsolete. It is important to note that there are a number of other products on the market which could be considered non-CASE tools, however the author feels that a valid sample of available products has been presented.

Choosing and Using a Methodology

The introduction to this paper makes the assertion that the majority of HP users do not currently utilize a structured methodology for developing systems. This assertion is based on discussions with HP users in a variety of industries at the various Interex conferences and through various other professional organizations. The primary reason the HP community has resisted the use of methodologies is that HP systems, unlike many other systems are easy to learn and easy to use. However, as HP systems become more sophisticated and as all systems become more "open", the MIS community will be looking for common platforms on which the hardware or architecture is transparent to the developer. As this evolution takes place, a standard development methodology becomes more critical. The MIS departments who already have chosen, and are in the process of using, standard methodologies will be in front of the curve.

The Process of Choosing

The process of selecting a methodology is similar to selecting packaged software for end users. The steps include:

1. Get upper management to "buy in"
2. Analyze the users' needs
3. Research available methodologies
4. Choose methodologies that most closely match users' needs
5. Get user commitment
6. Contact other sites using the methodology
7. Negotiate Contract (if applicable)
8. Train users
9. Implement methodology

Sometimes choosing a methodology can be easy because some users (in this case designers and/or analysts) may already be using some of the techniques described above.
The most difficult problem may be situations where two or more analysts have worked with (and learned to love) different methodologies. Although it is not necessarily a "bad thing" for analysts to be familiar with a variety of methodologies, it is important to remember that one of the primary reasons for using a structured methodology is for consistency among and between systems.

Implementing a Methodology

Implementing a chosen methodology may be the most difficult task in the process. A methodology can be implemented without using an automated tool or any CASE product if these tools are unavailable or too costly. However, it makes little sense not to automate a set of predictable, structured processes that will be repeated and reused an infinite number of times.

The primary problem with using CASE products is that no one product currently covers all aspects of the Systems Development Life Cycle. Some of the products work well for the front end analysis and design phases while others are geared toward the back end development, coding and implementation phases. There is little doubt that in the near future we will begin to see products that cover all phases of the SDLC but until that time developers have two choices:

1. Choose a methodology and product that focuses on the weakest area of system development (this will vary from shop to shop) or,

2. Utilize two or more methodologies and products that focus on specific aspects of the development process.

Although many purists will vote for the first alternative and claim that most structured methodologies can be applied to all phases of the SDLC, the author feels that the second alternative is the most rational. Using different approaches for the various phases of the SDLC clearly defines the phase of development while enforcing standards within each phase. For example, Data Flow Diagrams can effectively be used for the user requirements, system specification and the preliminary system design phase. Structured English and the

use of a data dictionary can be used to supplement the system design phase. Nassi-Schneiderman Charts can effectively be used for program development, coding and preliminary system testing. The Analysts' Tool Kit (ATK) or Excelerator may be used to automate much of the development process. Other methodologies and tools may be combined for a more thorough approach. Action Diagrams may be supplemented with Prototyping. Speedware may used as the tool for automating the majority of the development process.

The three points that need to be made are that:

1. Methodologies save time and have been proven to produce a better product.
2. Methodologies should be used for their strong points.
3. Even though full CASE tools will be available in the near future, choosing and using a methodology should not be postponed.

Summary

The primary focus of this paper is on getting MIS Managers (specifically HP users) to focus on the need to choose and use a development methodology. Several methodologies are reviewed and available CASE tools are discussed. It is hoped that the reader will glean some basic information about how to go about choosing a methodology or a set of tools. This paper is not intended to be a comprehensive review of all available techniques or tools but rather a pointer to some valuable resources that may assist in the selection and implementation process.

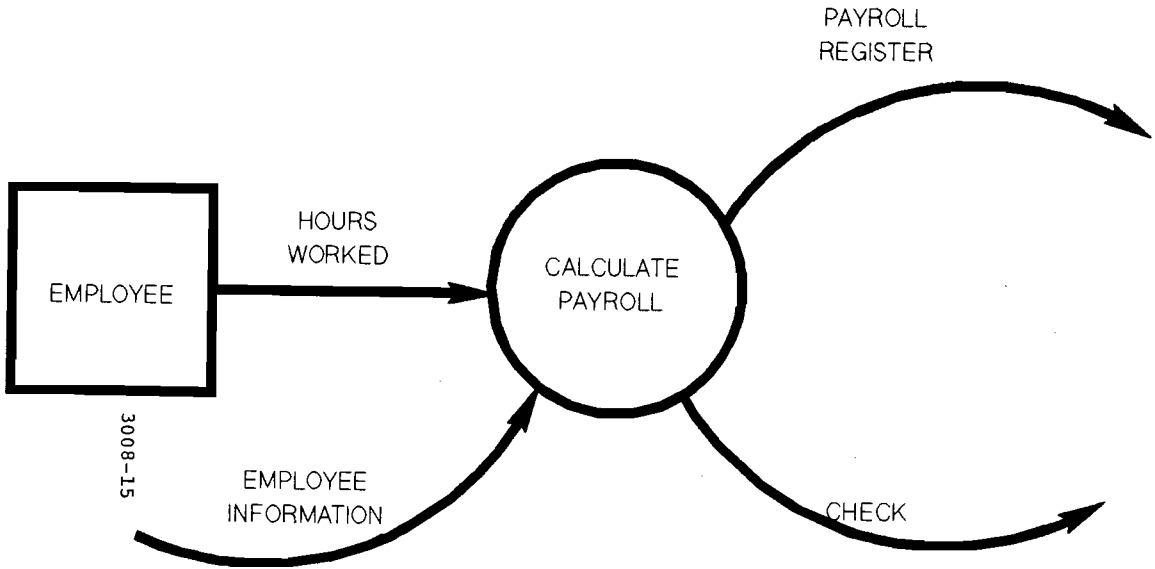
The reader should expect, just as with any other selection or implementation process, that potential pitfalls exist. There must be a management commitment to selecting and using a methodology. If not, such a project is doomed to failure. There will be a significant cost associated with the acquisition of a methodology or a tool in terms of research time and training. If these dollars are not made available, it is likely that the implementation will not be completed. A long-term commitment to using the techniques or the products is required. If these products are not used properly, the project will undoubtedly fail. Training, quality assurance, and refinement are all key elements of successful implementations of any structured methodology.

As Edward Yourdon says " what then is the purpose of having a methodology? There are three primary objectives:

1. To define the activities to be carried out in a

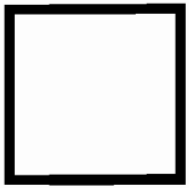
systems development project.
2. To introduce consistency among many systems development project in the same organization.
3. To provide checkpoints for management control for go/no-go decisions. " (1)

1. Yourdon, Edward, Modern Structured Analysis. Englewood Cliffs, N.J.: Yourdon Press. 1989, page 79.

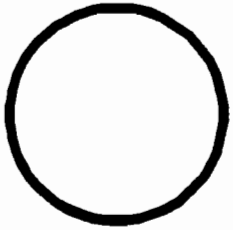

PAYROLL
REGISTER
EMPLOYEE
HOURS
WORKED
CALCULATE
PAYROLL
3008-15
EMPLOYEE
INFORMATION
CHECK

APPENDIX A-1

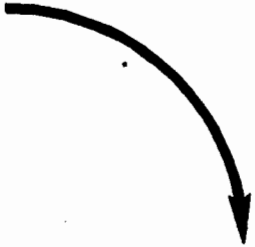
THIS ELEMENT REPRESENTS
A FILE OR A TEMPORARY
REPOSITORY OF DATA



THIS ELEMENT REPRESENTS
A SINK OR SOURCE WHICH
IS TYPICALLY A PERSON OR
ORGANIZATION

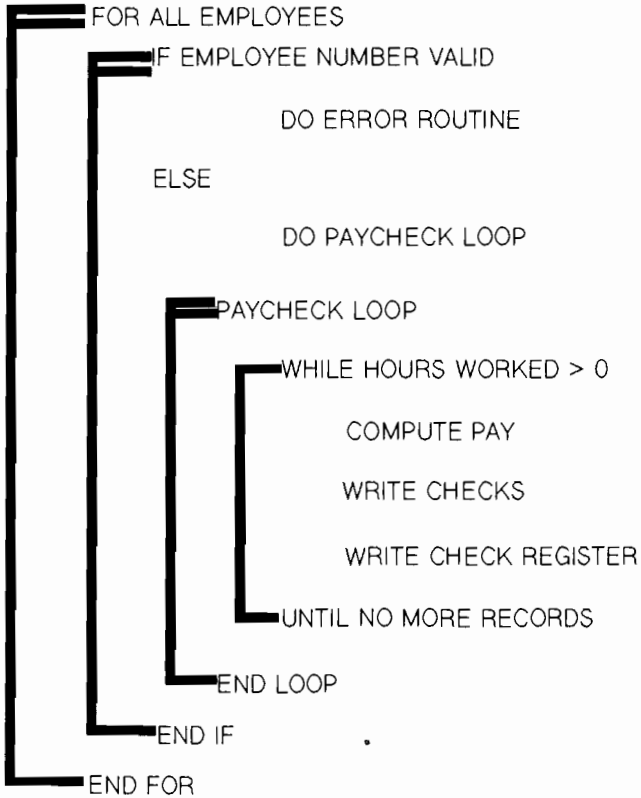


THIS ELEMENT REPERESNTS A
PROCESS OR TRANSFORMATION
OF INCOMING DATA FLOW



THIS ELEMENT REPRESENTS
THE FLOW OR PIPELINE THROUGH
WHICH PACKETS OF INFORMATION
ARE PASSED

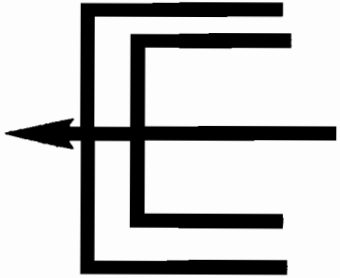
3008-16


FOR ALL EMPLOYEES
IF EMPLOYEE NUMBER VALID
DO ERROR ROUTINE
ELSE
DO PAYCHECK LOOP
PAYCHECK LOOP
WHILE HOURS WORKED > 0
COMPUTE PAY
WRITE CHECKS
WRITE CHECK REGISTER
UNTIL NO MORE RECORDS
END LOOP
END IF
END FOR

APPENDIX B-1

THIS CONSTRUCT DESIGNATES
ENTRY AND EXIT POINTS

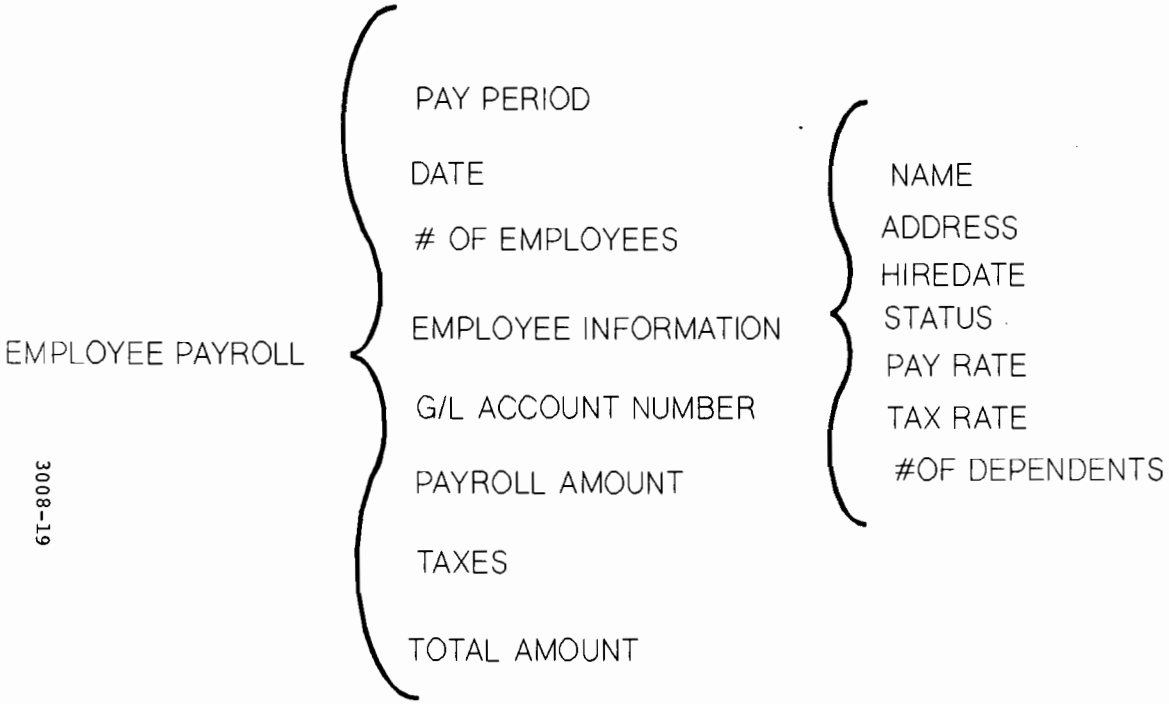
THIS CONSTRUCT DESIGNATES
REPITITION



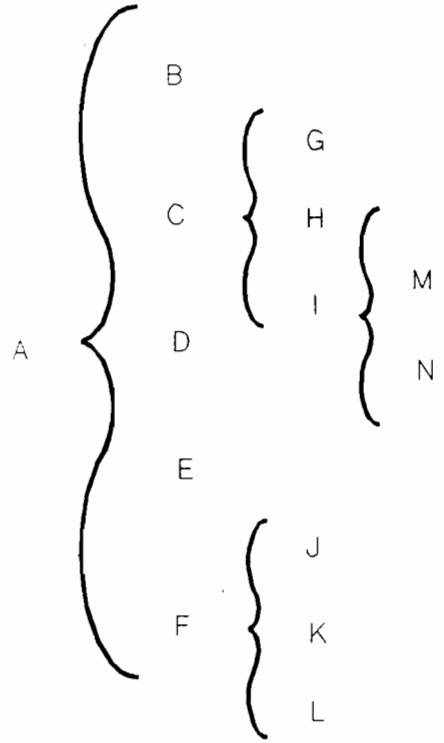
THIS CONSTRUCT DESIGNATES
ESCAPE POINTS

3008-18

APPENDIX B-2


EMPLOYEE PAYROLL
PAY PERIOD
DATE
OF EMPLOYEES
EMPLOYEE INFORMATION
G/L ACCOUNT NUMBER
PAYROLL AMOUNT
TAXES
TOTAL AMOUNT
NAME
ADDRESS
HIREDATE
STATUS
PAY RATE
TAX RATE
#OF DEPENDENTS
3008-19

APPENDIX C-1



3008-20

THIS CONSTRUCT UTILIZES BRACKETS TO ENCLOSE MEMBERS OF SETS. SEQUENCE, ALTERNATION AND REPETITION ARE SHOWN THROUGH THE USE OF THIS TECHNIQUE.

```
FOR ALL EMPLOYEES
        OBTAIN EMPLOYEE RECORD;
        IF EMPLOYEE RECORD IS VALID
           GET EMPLOYEE DETAIL RECORDS
           GET EMPLOYEE HEADER RECORD
        ELSE
           ISSUE INVALID EMPLOYEE MESSAGE
        END IF;
      FOR ALL ACTIVE EMPLOYEES
          OBTAIN HOURS WORKED RECORD;
          OBTAIN EMPLYOYEE INFORMATION RECORD;
            IF HOURS WORKED ARE GREATER THAN ZERO
               COMPUTE EMPLOYEE PAYROLL;
               UPDATE EMPLOYEE DETAIL REORDS
               WRITE EMPLOYEE CHECKS;
               WRITE CHECK REGISTER;
             ELSE ISSUE "NO HOURS WORKED" MESSAGE
            END IF;
      END FOR;
END FOR.
```

THE WORDING USED SHOULD BE AS EASY TO UNDERSTAND AS POSSIBLE- NOT IN "PROGRAMMESE".

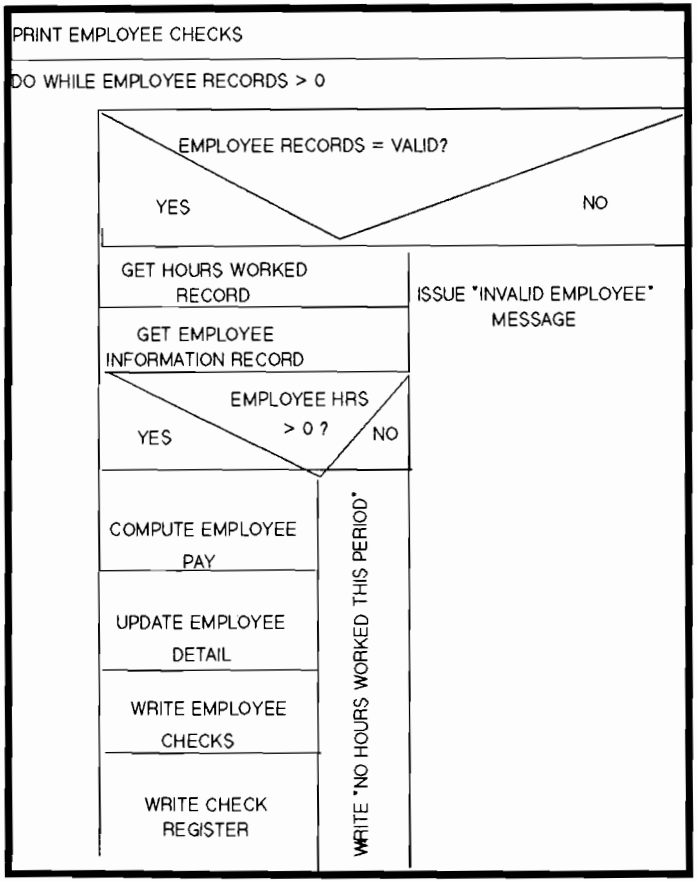
INDENTATION SHOULD BE USED TO SHOW THE PROCESSING HIERARCHY.

SEQUENCING, REPITITION AND CONDITION SHOULD BE MADE CLEAR THROUGH THE WORDING.

3008-22

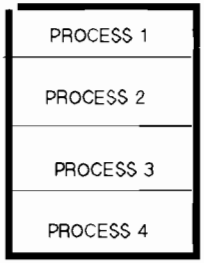
STANDARD KEYWORDS ARE USED FOR LOGIC; KEYWORDS SHOULD BE DEVELOPED AT EACH SITE AND SHOULD BE TIED TO ANY 4GL LANGUAGE OR ANY AUTOMATED TOOL USED AT THAT SITE.

TERMS LIKE "ENDIF", "END FOR" OR "EXIT" SHOULD BE USED TO CLEARLY SHOW WHERE THE STRUCTURE ENDS.

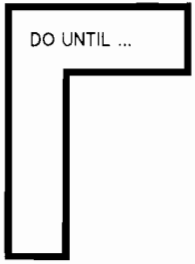

PRINT EMPLOYEE CHECKS
DO WHILE EMPLOYEE RECORDS > 0
EMPLOYEE RECORDS = VALID?
YES
NO
GET HOURS WORKED RECORD
ISSUE "INVALID EMPLOYEE" MESSAGE
GET EMPLOYEE INFORMATION RECORD
EMPLOYEE HRS > 0 ?
YES
NO
COMPUTE EMPLOYEE PAY
UPDATE EMPLOYEE DETAIL
WRITE EMPLOYEE CHECKS
WRITE CHECK REGISTER
WRITE "NO HOURS WORKED THIS PERIOD"

APPENDIX E-1

CONSTRUCTS FOR NASSI-SHNEIDERMAN CHARTS



THIS CONSTRUCT DENOTES
SEQUENTIAL PROCESSES.



THIS CONSTRUCT DENOTES REPITITION
SUCH AS DO WHILE OR DO UNTIL.

THIS CONSTRUCT DENOTES
DECISION, SELECTION OR CASE.

RECOVERY BY DESIGN

JAMES A. DEPP
UP TIME DISASTER RECOVERY, INC.
643 W. STADIUM LANE
SACRAMENTO, CALIFORNIA 95834
(916) 648-1282

The Need

The need is real - business functions are disrupted by interruptions in data processing services, and data centers are destroyed. Power service and data communications are interrupted, sometimes for days or weeks. Access is denied when contaminants or other hazards develop. Natural disasters do impair or destroy facilities. Human errors and deliberate sabotage do force a system down. The strategy presented is that of preparedness, but without any guarantee that such events can be avoided; they can not be. But without advance planning, their impact is much greater, their effect longer lasting and more devastating.

In a University of Texas, Arlington study 62.9% of the respondents had a recovery plan in place. Service outages had occurred at 31.6% of the facilities, in spite of plans, preventive measures, and security. The most common outages were utility, equipment, and people failures. But nearly 30% of these firms experienced liquid damage - floods, leaks, etc. Outages due to gases affected more than 10%. Extreme temperature from fire or steam leaks impacted more than 10%. And a combination of vandalism, sabotage, living organisms, projectiles, and earthquake affected 10%. Those 160 responding firms were 12-15% of the surveys sent out, and with the high percentage reporting plans in place, they appear to be actively involved in disaster prevention. Even if there were no incidents among those not responding, the figures suggest that an HP3000 is struck by disaster on a weekly, if not daily basis.

The Arlington study concluded that 85% of the organizations were heavily dependent on their data processing. Further, financial and functional losses increase rapidly with a disaster. In general, they concluded that organizations are not prepared. In spite of plans being cost effective, they are underutilized. We believe the plan is not complete without alternative processing capability, and it is not complete unless it is tested regularly.

Definition - Disaster

A working definition for disaster is 'an interruption resulting in extended or total loss to one or more operating facilities including data processing'. Determining the length of the interruption which can be tolerated is a part of the planning activity, and varies greatly among organizations.

Objectives to be Met

The first step is to agree on objectives of the business continuation strategy - recovery plan. This is an excellent time to gain upper management concurrence and commitment, a necessary prerequisite before continuing. In most cases, the recovery strategy is viewed as protection from such hazards as:

- Financial Loss or Reduced Profit
  - Cashflow interruption
  - Loss of customers
  - Reduced market share
  - Loss of investor confidence
- Service Interruption
  - Loss of competitive edge
  - Erosion of image
  - Confidence loss
- Liability or Legal Responsibility
  - Violation of contract
  - Violation of law

We must consider the people who could be affected:

- Shareholders and Financial Analysts
- Public
- Employees
- Customers
- Government / Authorities
- Media

How quickly the impact of a disaster will be felt must also be considered. Such assessment will require evaluation of multiple disaster scenarios - breakdown/repair, fire, flood, facility lockout, etc. The University of Texas used daily revenue as its reference, and looked at loss of revenue and added cost as a percent of revenue as a means of evaluation.

By plotting percent of revenue (y-axis) and time in days (x-axis), the impact becomes visible both in time and magnitude. High cost or quick impact demand quick response. Lower cost or more delayed impact provides additional options. The changing nature of data processing as it

becomes information management, accelerates the risk.

Increasing Significance of Disaster Recovery

As the role of computers and information evolve, so do their needs for protection and survivability. The progression of data processing might be shown as follows.

	Conventional/Historical Perspective	Advanced Status
Activity	Automate manual operation -faster, cheaper, fewer errors	Strategic
Method	Single processor or processors	Integrated, shared, distributed
People	D.P. professionals and central data staff	End Users - through and external to the organization

As the breadth of the information's use grows, so do the impact and cost of disaster. Where a delay previously might cause a late report, today a critical, time sensitive decision might be delayed. Previously, a failure might be felt by the data processing staff; now it may affect a group across the organization, or from bottom to top, or those outside the organization but dependent upon it.

Getting Started

There are three activities associated with providing business continuation - planning, testing, and actual recovery when needed. Each activity benefits from a stepwise approach. The setting of objectives and involvement immediately of top management are the starting points. Each activity should then be approached in a manner

- Phased - complete all that you can, not all that is possible.
- Structured - separate the activity to utilize others and designate workable pieces of the whole
- Planned for growth - minimize the immediate obsolescence which will frustrate the effort
- User involved - this is not a data processing effort, but a business effort

In planning, set overall recovery priorities, and then develop the plan in those priority tiers. Work with user teams to detail technical versus operational issues to be resolved, delegating operational issues to the user teams.

Make heavy initial use of a copier to get key information protected and offsite, rather than creating new text solving all the old problems at once. Refine the resulting material and strategies, deferring major revisions until after the inital recovery test.

For testing, start with easily defined, likely to be successful goals. It is the plans, not the individuals, being tested, and the testing is a part of plan development. Expect parts of the plan to need revision with each test, and plan a series of tests to confirm and solidify increasingly larger business activities. Always update the plan following tests.

Recovery will result from the planning and testing. Depending upon the size of the computer and data storage capacities, recovery may begin with only the most critical users and applications. These are defined by the plan. As replacement equipment arrives, increasing amounts of processing will be brought current. In structuring the recovery and phasing users and equipment and communications, balance is achieved between annual costs and costs at the time of the disaster.

Recovery Site Options

The objective for a recovery site is a suitable hardware environment for processing, and software and data to maintain services to users. This is not necessarily all software, all service, to all users, on a normal schedule.

Each of the recovery options has pros and cons. The options are:

- Manual Operation
- Fully Redundant System
- Agreements with Others
- Service Bureaus
- Vendor Facilities
- Hot Sites (Stationary, Mobile)
- Empty Shell

By reviewing the pros and cons of each option, tradeoffs and appropriate use can be identified. For example, a printer problem could be overcome through agreement with another HP user, but full recovery may require a hotsite. If the hotsite daily charge is high, a fallback empty shell may be good in combination for use when replacement equipment arrives.

Manual operation is possible for small systems, with low transaction volume, where in-house expertise still exists, and where data communication is not required. By contrast, redundant systems provide everything, but at a very high cost.

Agreements with others - reciprocals - also serve relatively small systems. Compatibility must be maintained, and capacity must be available quickly, usually requiring excess hardware. Written agreements are recommended. Service bureaus provide similar availability, without reciprocity.

Vendor facilities, if available, may also support processing on small, simple configurations. Large amounts of data would be unworkable, and testing would be more difficult.

Hot sites provide a shared redundant system, staffed for assistance, with systems in place for testing. Unique equipment can be installed, and security and administrative controls make it 'your' shop. Mobility adds the advantage of staying at home, with full staff, and data communications from your existing network.

Critical Processing

Small and medium sized systems can frequently be recovered in total given the sizes of typical backup systems. Large processing sites will have fewer backup options. Economics will influence the size of the backup configuration chosen, leading to the decision of what is most critical to the business survival. Even if all applications are to be recovered, the sequence in which they will be recovered and accessible must be determined.

We have seen good success with discussion/role play as a method of prioritizing activities. By gathering key people from each area together for discussion, and for part of the discussion assigning to each the support for another's position, consensus can be achieved.

Additionally, the approach using percent of revenues versus time can also provide visual clues which may support the group consensus. It must be remembered that the particular time of week, month, or year when the disaster occurs will impact the priorities as well. A school district will be less concerned with the budget at registration time, and less with transcripts at budget time. At key times, any one application might be critical or low priority.

The Recovery Team - People

In time of need, help is usually appreciated. But organized and trained help is not always available. With generally small data processing staffs, it is prudent to develop and organize the help in advance, and to draw whenever possible on resources which enlarge the team. Identify, organize, and train these people now.

The specific recovery organization will fit with the larger corporate structure. Key team leaders and alternates are first assigned. Three key leaders are the Recovery Plan Coordinator, the System Recovery Manager, and the Rebuilding Project Manager. If workable in your organization, have key support people assigned by purchasing, personnel, legal, accounting, and other staff groups. These people will help the functional activities of establishing an alternate site, recovering data, system recovery, operations recovery, training, etc.

The plan should describe roles and tasks, and members of the teams should be identified by position or function. Names of the individuals should be separately maintained as a list which can be easily updated. Chronological and check lists should be provided. Communication lines should be clearly defined.

Spending needs and limitations should be identified in the plan, having been established in the planning process and preapproved. This is another opportunity to reinforce upper management involvement and commitment.

As a part of the plan, include the assumptions, economic and other conditions upon which the plan is based. This makes it much easier later to identify change, or to reevaluate previous assumptions.

Procedures, Prevention, and Recovery

Procedures are established to facilitate any activity. They are foundational to the smooth running operation. The next level of defense against interruption is prevention. Halon for fire, sensors for water, proper building design, security are all preventive measures, and desireable. But none of these can guarantee that a disaster will not occur. While many interruptions occur due to breakdown in these measures, others are externally caused - the hurricane, the tornado, the flood, the earthquake.

Hardware that is standardized, kept at current revision levels, and which is well documented as to interfaces, strap and switch settings, and wiring is easier to duplicate quickly. By choice, avoid the antique, the unique, and the ultra-new; recognize the risk of slow replacement when these can not be avoided. Recovery priority may be impacted by this availability of hardware.

Data should be structured to optimize daily operation and allow efficient recovery. These two are not contradictory, and benefits have been achieved by data reorganization in some cases. Documentation is important, as is review of the flow of the data. Are source documents at risk in disasters or will updating of backup data be difficult or impossible?

If so, what can be changed? If sharing of systems is planned, naming conventions must be established and monitored.

Software design should anticipate the recovery hardware options, should be well documented, and all components should be well identified.

The computer facility should be designed to enhance recovery even though fire may destroy the building. Separation of computer, datacomm, and supplies might help. Standard utility components will speed reconstruction. Preventive measures can reduce damage.

Data communications lines can be protected. Equipment can be standardized to provide redundancy for critical locations. Compatibility can be maintained so that modems can be interchanged. Spares and strapping documentation should be stored offsite.

With increasing distribution of computing, the personal computers, LAN's and servers, and word processing systems need to be considered for backup. Their data may be equally critical, and backup methods may be automated to protect the data.

Microwave and satellite communication may need to be preserved. Voice communication should not be overlooked, and the sophistication of current digital PBX units makes them just one more computer. The scope of recovery planning can grow beyond reasonable limits, or organizational bounds. These resources are mentioned primarily to cause evaluation, and hence appropriate assignment of tasks.

Phased Development

Providing recovery opportunities is not a small task, but it is a manageable one, just like the large development and programming tasks which have have been accomplished before. The diversity of the task is sketched above to show at the outset where the activity might lead, but the emphasis should be on stepwise, phased development of strategy and implementation. Otherwise, too often, the task is never started, and the disaster reminds everyone of the need.

A 950 MIGRATION PROJECT

Lewis R. Dalrymple
City Of Tempe
120 E. 5th Street
Tempe, Arizona 85281

There have been many excellent papers written on the 'Migration Process' from numerous points of view. I would like to add to these a 'real time' migration experience from the first 900 series migration at the City of Tempe.

INTRODUCTION

It is most important that we properly set the stage for the project I will describe in this paper.

Unlike many migration projects, our migration included many 'facets' other than a 950 series computer.

We have a multi-computer facility comprised of 9 HP3000 computer systems providing services to 30 departments throughout our municipal complex.

Our ongoing 5-Year Plan allows us to use computers, which are replaced by a new purchase, to upgrade another computer system in need of better processing capabilities.

As part of this plan, a series 70 currently being used in our administrative area was scheduled to be upgraded to a 950 and the series 70 would in turn replace a series 58 used for our library system. In both cases the respective computers were suffering from acute performance problems.

The time was right for our two system migrations to take place, and as quickly as 'practical'.

THINGS TO CONSIDER

The primary system being migrated (series 70) provides all processing for the City administrative functions, including financial, utilities, permits, and all office automation needs (HPWORD, HPDESK, VISICALC, LISTKEEPER, HPDRAW):

* 22 Applications
* 240 Users (50 HPWORD Users)
* 2 Locations
* All RS232 Direct Connected

With this criteria we determined several key items which affected our migration:

* Computers must be interconnected with a Local Area Network (LAN)--No DS on the 950.....
* HPWORD (word processing) users must be converted to personal computer (PC) based software and connected to the LAN for interface to the 950 and/or other systems.
* City wide LAN/WAN must be designed and installed to provide appropriate user/systems interconnections.
* Software applications must be modified for compatibility/ native mode to run on the 950.
* Staff/users must be trained in PC, LAN, and 950 use/ operation.
* All user/application interdependencies must be carefully considered to determine when and how they could be migrated with the least disturbance to the end user.

At this point it was very obvious that some form of Project Management was needed to assure a successful and timely migration.

THE WORKBOOK APPROACH

Having managed many projects over the years, I decided to use a tried and trusted method -- The Workbook Approach.

The Workbook Approach to Systems Analysis and Project Management is not new and certainly is not my creation. I first became aware of this approach in the late 1970s when I used the Systems Analysis Workbook Guide, developed by Robert D. Carlsen and James A. Lewis, in developing software systems.

I further refined their idea, and have been using it ever since.

TOOLS FOR PROJECT MANAGEMENT

The key to the Workbook is the standardized worksheets which provide both direction and documentation for your project. I call them 'The Tools For Project Management'.

We will use these worksheets to specify, plan, schedule, implement, and control our project.

The Workbook is divided into seven (7) project areas:

1) The Project Specification
2) The Project Plan
3) Task Planning
4) The Project Team
5) Task Assignment/Scheduling
6) Resource Planning
7) Implementation And Control

PROJECT SPECIFICATION

Our specification was dictated by both our 5-Year Plan and the computer we elected to use in our next migration step. The general specification included:

a) A series 58 would be upgraded to a series 70.

b) A series 70 would be replaced with series 950.

(As a result of (b) the following items were dictated.)

c) Computers/users must be interconnected by a LAN.

d) HPWORD users must be converted to PCs.

e) LAN must be designed and installed.

f) Staff/users must be trained.

g) Applications softwre must be modified.

h) All users/applications interdependencies must be determined.

PROJECT PLAN

I guess the best way to state this is really,

PLANNING THE PROJECT .....

Project Management is the process by which it is assured that the objective is achieved and resources are not wasted.

Planning is one of the two essential parts of Project Management. Control is the other.

Probably the most neglected area in design/development involves planning and control of the project. More than one disastrous project has been launched by 'computer people' who communicated their aims to unknowing users using DP 'buzz words' in lieu of specific lists of easily understood tasks, schedules, and costs.

Any project must first be planned in detail. Control is involved with comparing actual progress with the plan and taking corrective actions when the two do not correspond. Without the plan, true control is not possible; the need for corrective action, its nature, extent, and urgency cannot be accurately determined.

No matter what the magnitude of a project, it should be planned and controlled. This does not mean, however, that the same level of planning effort should go into projects, regardless of size.

The Project Plan must include the following:

* Definition of the objective(s) or product goal.

* Definition and description of tasks needed to be performed in order to meet the objective(s).

* Determining interdependencies and interrelationships of various tasks, and the scheduling of these tasks.

* Determining the resources required to perform the tasks.

* Logically grouping the tasks and assigning them.

* Pricing and budgeting the tasks.

Our project plan was developed as follows:

* Overview

  The overall migration plan is uniquely dependent upon the users and the services they expect from the computer system/software. This is to say, that if equal services are not available on the new system, a migration cannot take place.

  With this in mind, we divided all users affected by the migration, into 5 types (groups):

User Types

Group - 1

No specific dependencies and can migrate to 950 whenever equipment is available.

Group - 2

Office Automation Software/LAN Dependent

Group - 3

Application Dependent

Group - 4

LAN Dependent

Group - 5

O/A-SW, Application, and LAN Dependent

These groups vary from no specific dependencies (other than equipment/software installation) to a wide range of single or interdependent needs.

Anyone and/or combination of these dependencies will then dictate when and under what conditions a particular user, application, or system may be upgraded or changed.

Our plan, therefore, provides for several phases or migration steps to allow for all user types.

* Equipment Migration

The equipment migration (system hardware/software components) will take place in 6 phases or steps:

1. Existing computer network (DS) will be upgraded to a System LAN.

2. System LAN will be extended to City Hall complex to provide links to user STARLANS.

3. User STARLANS will be installed for selected "early" migration users. (Limited)

4. Series 950 computer will be added to System LAN.

5. DTC (Distributed Terminal Controllers) will be added to the System LAN.

6. Additional STARLANS will be added to the System LAN.

* User Migration

User migration will also occur in several steps as follows:

1. "Early" migration users will be selected, trained, and connected to the STARLANS using Series 70 resources.

2. "Early" migration users will be migrated to the Series 950.

3. DTC (Group - 1) users will be migrated to the Series 950.

4. Balance of users will be migrated to the Series 950 as STARLANS, TRAINING, SOFTWARE, and/or other dependencies are resolved.

* Applications Migration

Applications migrations will take place in 3 phases:

1. Analysis of applications software code on Series 70.

2. Modification/updating code as determined by the analysis phase.

3. Installation/testing of new code on Series 950.

TASK PLANNING

There are several important considerations involved in breaking down tasks required to do a job.

The first consideration is that each task should have a unique control (task) number. Once assigned, these numbers will carry through and serve as a cross-reference for the same tasks in all project documents, including schedules, lists, assignments sheets, etc.

Any numbering scheme that is convenient and easy for the Project Manager or task leader may be applied. However, it should be a scheme as to allow for later additions or deletions of sub-tasks without the necessity of renumbering.

The next important aspect in task planning is that each task title denotes an actual element of work.

Once tasks have been broken down and defined, task statements (statements of work) can be generated.

With tasks satisfactorally defined, the next important planning effort involves determining the interdependencies and sequences of the tasks. A useful technique is the network. In network terminology, circles usually represent events, while lines represent tasks. We provide several examples.....

We will use the PERT (Program Evaluation & Review Technique) system to demonstrate the network.

There are three basic steps in creating and using the network:

1) Develop the network of events.

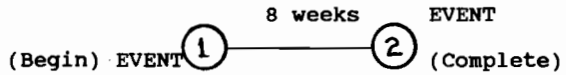
2) Estimate the time to accomplish each activity (task) in the network.

3) Analyze, through simple calculations, the schedule problems in getting from event to event.

Step 1 - Rules

i) An event is a specific accomplishment required.

ii) The event must be recognizable as occurring at a specific point in time.



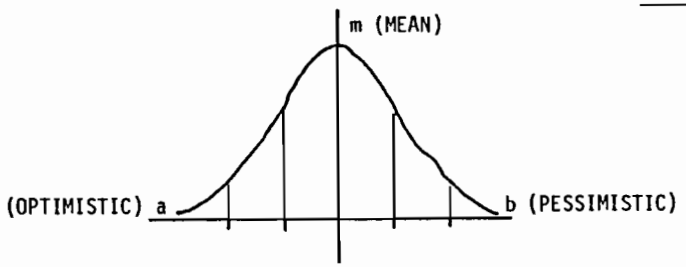
Step 2 - Rules

i) Define Optimistic Time (a)

ii) Define Pessimistic Time (b)

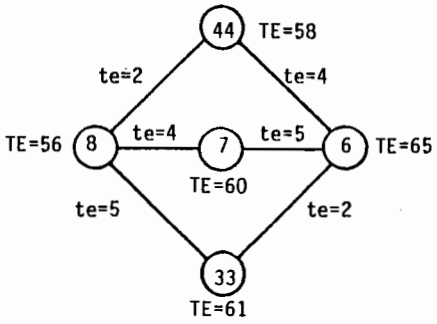
iii) Compute Most Likely Time (te)

$$te = \frac{a + 4m + b}{6}$$



Step 3 - Rules

i) Determine the amount of slack associated with various activities (this will define critical path).



The longest time (56+4+5)=65

ii) Make resource/time adjustments.

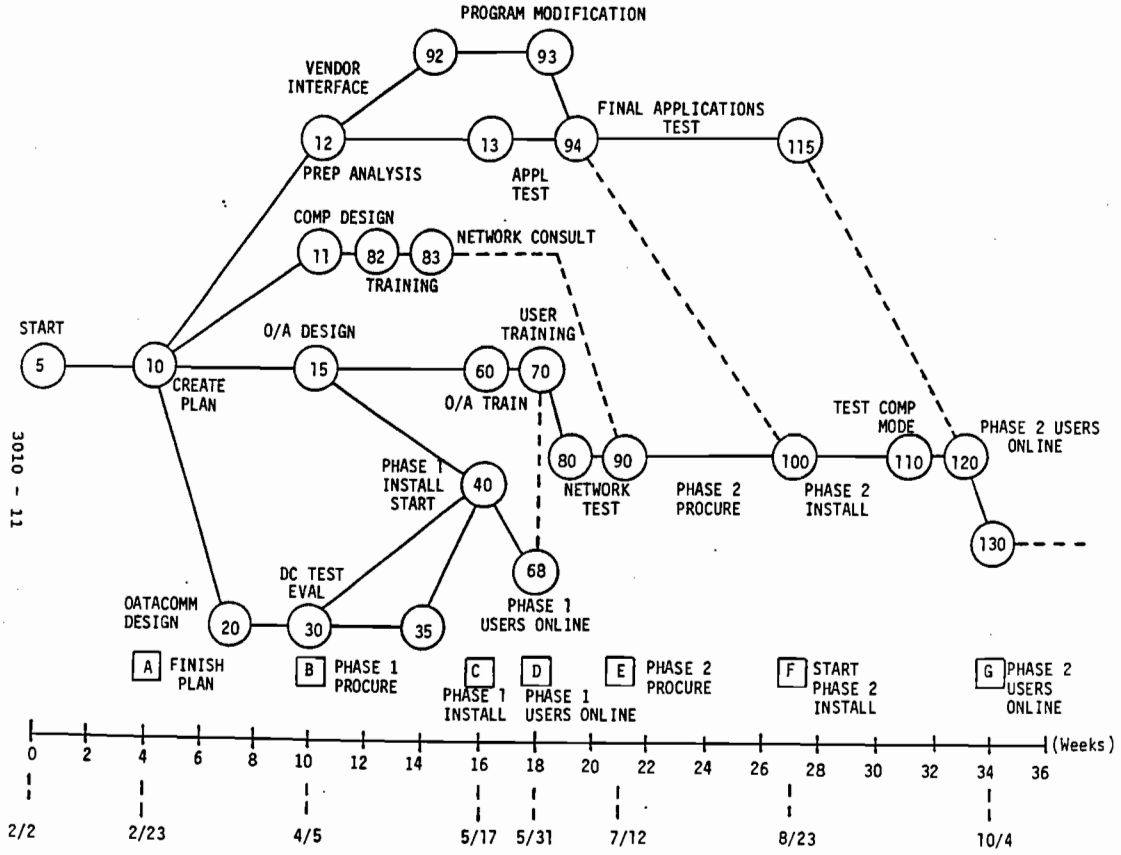
From this information, we developed the basic network diagram of our project. See figures NW-1 and NW-2.

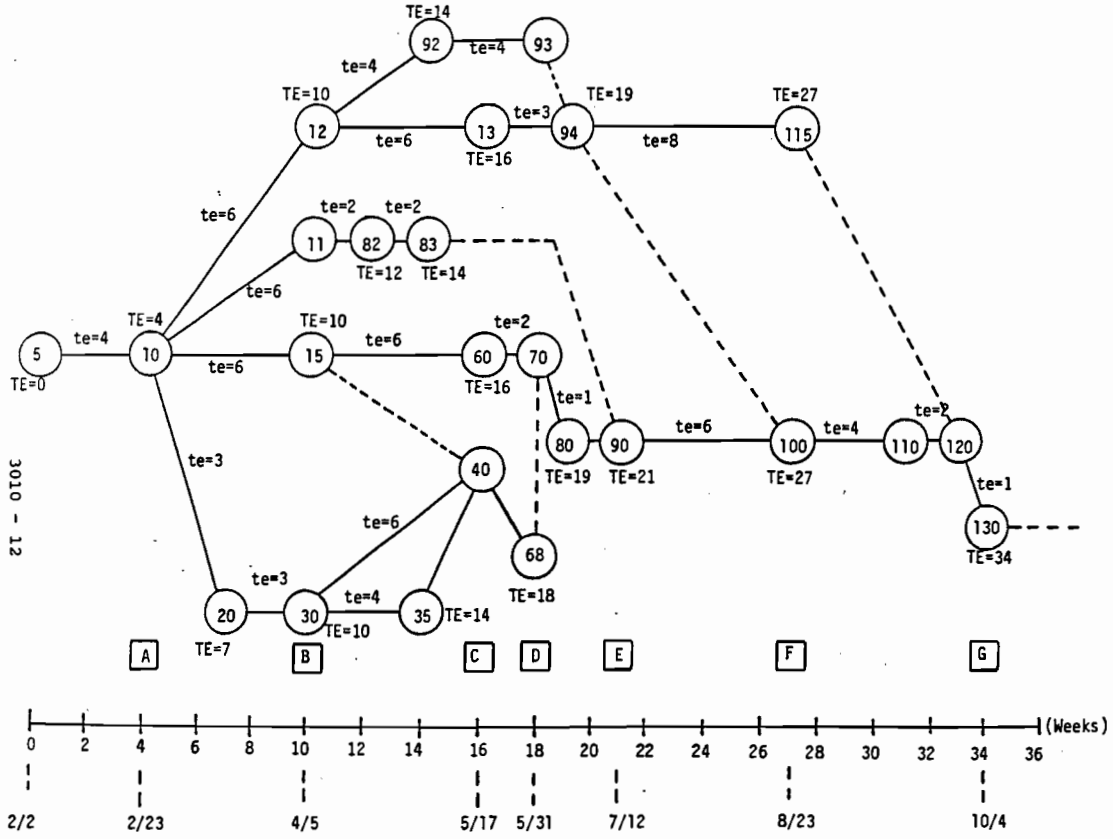
Figure NW-1 shows our network starting with event (5) and ending with event (130). As you can see, the network diagram describes each major event for our migration project.

Figure NW-2 provides us with our estimated times to accomplish each task (te) and time (in weeks) when an event is started and/or completed (TE). As you can see, the major portion of our project was accomplished in about 34 weeks.

CREATE THE PROJECT TEAM

The creation of a project team can take place anytime before the actual task assignment and schedule step.


PROGRAM MODIFICATION
92
93
VENDOR INTERFACE
12
13
94
FINAL APPLICATIONS TEST
115
PREP ANALYSIS
APPL TEST
COMP DESIGN
11
82
83
NETWORK CONSULT
TRAINING
START
5
10
CREATE PLAN
O/A DESIGN
15
60
70
USER TRAINING
O/A TRAIN
80
90
NETWORK TEST
PHASE 2 PROCURE
100
TEST COMP MODE
110
120
PHASE 2 USERS ONLINE
130
PHASE 1 INSTALL START
40
68
PHASE 1 USERS ONLINE
DC TEST EVAL
OATACOMM DESIGN
20
30
35
A FINISH PLAN
B PHASE 1 PROCURE
C PHASE 1 INSTALL
D PHASE 1 USERS ONLINE
E PHASE 2 PROCURE
F START PHASE 2 INSTALL
G PHASE 2 USERS ONLINE
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 (Weeks)
2/2
2/23
4/5
5/17
5/31
7/12
8/23
10/4


TE=14
92
te=4
93
te=4
TE=10
12
te=6
13
te=3
94
TE=19
te=8
115
TE=27
TE=16
te=6
te=2
te=2
11
82
83
TE=12
TE=14
te=6
TE=4
TE=10
te=2
5
te=4
10
te=6
15
te=6
60
70
TE=0
TE=16
te=1
80
90
te=6
100
te=4
110
120
te=1
TE=19
TE=21
TE=27
130
TE=34
te=3
40
te=6
68
TE=18
te=3
te=4
20
30
35
TE=14
TE=7
TE=10
A
B
C
D
E
F
G
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
(Weeks)
2/2
2/23
4/5
5/17
5/31
7/12
8/23
10/4

3010 - 12

Although there may be several levels of project participants, it is essential that we determine the appropriate task leaders/managers who will provide both guidance and local control of any group of interdependent tasks.

These key individuals are normally determined in the task planning step.

Typically the breakdown could include systems (hardware), software, communications, etc.

The project team includes all staff members who will contribute to the project.

The Project Manager sets the major task assignments for the project team, and both the Project Manager and task leaders assign tasks to other subordinate personnel.

Project team members perform tasks according to a published project schedule and report their activities regularly.

Project members may work on the project full time or part time for either a short time or for the complete project.

In some cases, contracting and/or consulting personnel can be employed to provide needed assistance.

Once a project begins, the Project Manager will keep the 'task leaders' informed of all changes in task requirements and schedules. This will allow the task leaders to coordinate the activities of their respective staff members.

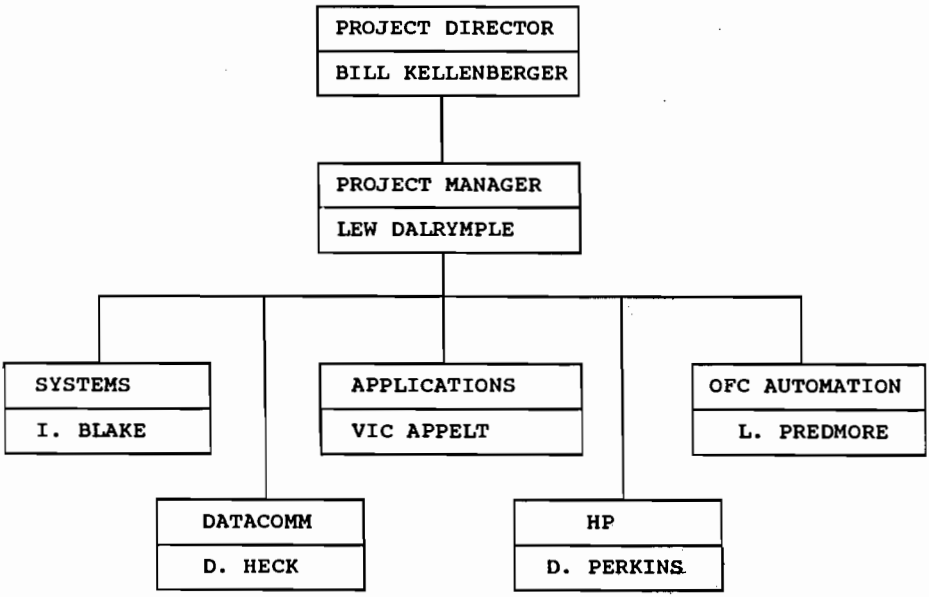
* The Project Team

  The project team consisted of personnel from the City of Tempe and Hewlett Packard.

  We developed a "Project Team" functional chart which describes the "major project players" or task leaders who managed specific areas of the overall project.

  Additional personnel were added when needed and they were managed by appropriate task leaders.

CITY OF TEMPE
950 COMPUTER SYSTEM
MIGRATION PROJECT TEAM



ASSIGN/SCHEDULE THE TASKS

Once the network is complete and the project team is in place, task assignment and control schedules can be developed.

Input for schedules include the network, project specifications, and task planning data. The scheduling of tasks should normally be done in conjunction with the planning of resources.

The process may take several sub-steps in order to provide the best estimate of time/resources to accomplish all of the assigned tasks.

* **PROJECT MANAGEMENT GROUPS**

  We first determined our MAJOR MANAGEMENT GROUPS. In our case, we decided upon six groups (1 thru 6):

  1 - Computer System

  2 - Applications (Programming)

  3 - Office Automation

  4 - Data Communications

  5 - Systems (Hewlett Packard)

  6 - Office Automation (Hewlett Packard)

  We then assigned one or more major group(s) to each project team leader.

* **ASSIGNING THE TASKS**

  We now are ready to assign our MAJOR TASKS, which we determined in the Task Planning phase.

TASK PLANNING

TASK NO.	PLANNING/ASSIGNMENT DESCRIPTION	ASSIGNMENT	DATE
A	PREPARATION/ANALYSIS	I. BLAKE	
B	COMPUTER SYS SPEC/DESIGN	I. BLAKE	
C	APPLICATIONS SOFTWARE	V. APPELT	
D	DATACOMM SPEC/DESIGN	D. HECK	
E	OFC AUTOMATION SPEC/DESIGN	L. PREDMORE	
F	PROCUREMENT	L. DALRYMPLE	
G	TRAINING	L. DALRYMPLE	
H	CONSULTATION	L. DALRYMPLE	
I	INSTALLATION/TEST	I. BLAKE	
J	CONVERSION/TESTING	I. BLAKE	
K	SUPPORT		

PROJECT NAME #950/89	DATE	PAGE
950 MIGRATION	2/25/88	
TASK GROUP ASSIGNMENT(LEADER)		

ASSIGN/SCHEDULE THE TASKS

Once the network is complete and the project team is in place, task assignment and control schedules can be developed.

Input for schedules include the network, project specifications, and task planning data. The scheduling of tasks should normally be done in conjunction with the planning of resources.

The process may take several sub-steps in order to provide the best estimate of time/resources to accomplish all of the assigned tasks.

* **PROJECT MANAGEMENT GROUPS**

  We first determined our MAJOR MANAGEMENT GROUPS. In our case, we decided upon six groups (1 thru 6):

  1 - Computer System

  2 - Applications (Programming)

  3 - Office Automation

  4 - Data Communications

  5 - Systems (Hewlett Packard)

  6 - Office Automation (Hewlett Packard)

  We then assigned one or more major group(s) to each project team leader.

* **ASSIGNING THE TASKS**

  We now are ready to assign our MAJOR TASKS, which we determined in the Task Planning phase.

TASK PLANNING(ASSIGNMENT MATRIX)

TASK NO.	DESCRIPTION	TASK ASSIGNMENTS 1	2	3	4	5	6
A -	PREPARATION & ANALYSIS						
1	SYSTEMS	X				X	
2	APPLICATIONS	X	X			X	
3	NETWORKS	X			X		
4	PERIPHERALS	X			X	X	
5	O/A SOFTWARE			X			X
6	O/A HARDWARE			X			X
B -	COMPUTER SYS SPEC/DESIGN						
1	COMPUTER	X				X	
2	DTC	X			X	X	
3	LAN	X			X	X	
4	SITE	X			X	X	
5	PT-TO-PT	X			X	X	
6	SOFTWARE	X				X	
7	TRAINING	X				X	
C -	APPLICATIONS SOFTWARE						
1	INHOUSE	X	X			X	
2	3RD PARTY	X	X			X	
3	SUPPORT/DEVELOPMENT	X	X			X	
4							
5							

PROJECT NAME	DATE	PAGE
950 MIGRATION	2/24/88	1 OF 4
MAJOR TASKS		

1 = SYSTEMS
2 = APPLICATIONS
3 = OFFICE AUTOMATION
4 = DATACOMM
5 = SYSTEMS(HP)
6 = O/A(HP)

PROJECT # 950/89		MILESTONE SCHEDULE (WEEKS)																																	
950 MIGRATION PROJECT			FEB				MAR					APR				MAY					JUN				JUL					AUG			SEP		
		0									1										2										3				
TASK NO.	TASK DESCRIPTION	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4
A -	PREP/ANALYSIS																																		
1	SYSTEMS			>>	>>	>>	>>	>>	>>																										
2	APPL'S SOFTWARE			>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>																					
3	NETWORKS	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>						
4	PERIPHERALS			>>	>>	>>	>>	>>	>>																										
5	O/A SOFTWARE			>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>																					
6	O/A HARDWARE			>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>																					
B -	COMP SYS DESIGN																																		
1	COMPUTER	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>																						
2	DTC	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>																						
3	LAN	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>						
4	SITE				>>	>>	>>	>>	>>	>>	>>	>>	>>																						
5	PT-TO-PT				>>	>>	>>	>>	>>	>>	>>	>>	>>																						
6	SOFTWARE				>>	>>	>>	>>	>>	>>	>>	>>	>>																						
7	TRAINING							>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>								
C -	APPLICATIONS S/W																																		
1	INHOUSE												>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>								
2	3RD PARTY												>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>								
3	SUPPORT/DEVELOP'T												>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>	>>								

RESOURCE PLANNING

Once the network tasks have been defined and sequenced, the job of estimating manpower and material requirements for the project is the next order of business.

Manpower and other resources required to accomplish the defined implementation tasks are developed with the Resource Planning worksheet. The format is similar to the Task Planning sheet.

In using the Resource Estimating worksheet, manpower estimates can be identified in terms of individuals or in terms of skill types.

We elected to record resources by major task groups (1 thru 6) and in manweeks of time.

IMPLEMENTATION AND CONTROL

Regardless of the planning excellence, there is one overriding consideration. The credibility of the entire plan, and the task leaders ability to perform to that plan, will be judged by managements' observance of the task leaders performance during the earlier analysis and design portion of a project.

It is, therefore, important that continuous feedback and feedforward information be supplied to control the overall project.

* Establish regular intervals (typically monthly) for overall project meetings with your project team, providing updated and field input as to progress/change of the project.

* Have individual meetings with each task leader (weekly) giving special attention to detail/progress of each task in the respective group.

* Provide regular updated documentation to all members, which includes revised task lists/descriptions, GANTT charts indicating critical path, and overall PERT charts of the project.

RESOURCE PLANNING

TASK NO.	DESCRIPTION	RESOURCE MANWEEKS 1	2	3	4	DATE
A -	PREPARATION & ANALYSIS					
1	SYSTEMS	3				3/20/88
2	APPLICATONS SOFTWARE	5	11			4/30/88
3	NETWORKS	3			18	6/05/88
4	PERIPHERALS	3			1	3/30/88
5	OFFICE AUTOMATION SOFTWARE			5		4/30/88
6	OFFICE AUTOMATION HARDWARE			5		4/30/88
B -	COMPUTER SYSTEM SPEC/DESIGN					
1	COMPUTER	5				4/30/88
2	DTC	2			2	4/30/88
3	LAN	1			6	6/05/88
4	SITE	2			2	4/30/88
5	PT-TO-PT	1			1	4/30/88
6	SOFTWARE	5				4/30/88
7	TRAINING	3				7/30/88
C -	APPLICATIONS SOFTWARE					
1	INHOUSE	5	15			8/10/88
2	3RD PARTY	5	15			8/10/88
3	SUPPORT/DEVELOPMENT	5	15			8/10/88

PROJECT NAME	DATE	PAGE
950 MIGRATION	2/24/88	1 OF 4

1 = SYSTEMS
2 = APPLICATIONS
3 = OFFICE AUTOMATION
4 = DATACOMM

* Be alert to any and all problems that arise, and make quick decisive corrections whenever necessary.

* Resolve any conflicts immediately.

* Stick to the plan/schedule.

Are You Missing the Boat? Prototyping Revisited

B. Bliesner, D. Knight, M. Michalak, C. Abbott,
J. Dixon, K. Eriksen, M. Fish, E. Yoneoka
Boeing Computer Services
Box 24346 M/S 6H-66
Seattle, WA 98124

Abstract

With expenditures on computing resources growing faster than other segments of companies' operating costs, increasing productivity in application development becomes all the more critical. Many have turned to Application Prototyping as a methodology for increasing programmer productivity and shortening application development time over traditional life cycle methods. There is a tendency to accept this methodology as static. However, in the past few years the computer industry has undergone tremendous change. New hardware and software products have appeared at a phenomenal rate. The concept of application prototyping needs to be updated to reflect these industry changes. "Prototyping Revisited" takes a practical look at the subject and describes the new tools, new technologies and industry trends on this continuously evolving methodology. This paper reviews past influences and puts prototyping in perspective for today's thinking. Now, more than ever, prototyping has a significant role in the data processing industry.

Introduction

'Prototyping', 'CASE', 'Expert Systems', '4GL', 'Neural Networks',.... The list of acronyms and 'buzzwords' associated with current computer applications technology is long. It can be confusing. At times it is beneficial to take some time to see where we've come from, to look around at where we are, and to see where we may go from here.

In 1986 Boeing Computer Services - Aerospace Support formed a team to study the concept of Application Prototyping on the HP3000. The team's goal was to research methods of increasing programmer

productivity and shortening application development time. To accomplish this goal the team began to investigate productivity tools, project control methodologies and the attitudes and philosophies of users, programmers, and their management. The team, representing more than 100 years of combined programming experience, came to grips with the litany of computerese which swirls around the world of Application Prototyping. They all knew a bit from a byte, Pascal from Fortran, but did they know the difference between CAD and CASE, a 4GL and an SQL? Books and articles were read; definitions pinned down. Eventually a sense and an understanding of this method of computing began to emerge. (Most of those who had worked in the Mini/Micro environment for any length of time discovered that they had been doing at least some aspects of application prototyping all along - they just didn't know it.)

This paper presents the findings of that team. It defines the Application Prototyping methodology, and then describes the types of tools currently available for Application Prototyping (here, among other things, is where the difference between a 4GL and an SQL will be revealed). Next it explores existing methodologies. Finally, areas of concern and future directions are discussed.

In this paper several software products are mentioned as examples of a particular type of tool for Application Prototyping. Mention of these products does not constitute endorsement by the authors or the Boeing Company. Lack of mention of a product does not indicate that it would not be suitable as a tool for Application Prototyping.

Overview/Definition

To get a sense of what Application Prototyping is all about, imagine a programmer working alone on a program which will require some kind of data entry and some kind of output based on that data. The programmer knows what must be accomplished. His[1] preparation prior to code and test probably includes flowcharts, data requirement lists, etc. After the first successful run of the program, he will view the screen or printout critically, looking at such things as accuracy, consistency, ease of use and presentation. Unless it is an extremely simple program, something was probably overlooked, or the format of the presentation isn't quite right. So he modifies the program until he is finally satisfied with it. Once everything is to his satisfaction, the application continues to run, supplying new or changed input and/or

fresh output whenever it is needed. And, if the application is used for any length of time, he will inevitably discover that at least one requirement, either for input or output, will change. After an analysis, the changes are incorporated into the existing program. Hopefully, our programmer had 'documented' the original program with internal comments and was able to find the flowcharts, etc. which aided in the program's creation. Testing and implementation take place and the program runs until a change in requirements comes along or until the need for it disappears.

Several steps in the development process for this program have occurred here:

- Requirements were established.
- Data requirements lists, flowcharts, etc. were developed.
- Program code was written and tested.
- Program flow and data entry routines were critiqued.
- Output accuracy and format were critically analyzed.
- The program was put into use.
- Modifications were made as requirements changed.

These steps are essentially the same for any software application, no matter what methods are used to generate a final product. The difference between Application Prototyping and more traditional methods of software development is the level of involvement of users in the development process. Traditionally, the end user doesn't get to use an application until it has been formally released to him, which may be months or years after the need for the application was established. The application meets the requirements, but in reality may not function as the user actually envisioned. Then too, lack of interaction with the end user can result in miscommunications. Errors present in initial phases have gone undetected. In traditional life cycle, the correction of these errors becomes more expensive as you move further into the life cycle of the application. In contrast to the traditional approach, in Application Prototyping only fundamental goals and objectives are determined. A working model of the application is quickly built with an advanced programming language (4GL). An iterative process then takes place between the analyst and the end user in which revisions and enhancements are implemented. This rapid interaction between analyst and end user is made possible by new technologies. A control prototyping methodology is necessary to complement these new technologies.

Application Prototyping is defined as follows for this paper:

> Application Prototyping is a methodology using tools and techniques to design, develop, and implement a computer application. As user requirements change or technologies improve these steps are repeated through the life of the application. This is accomplished through:

- Increased user involvement in the incremental development of applications.
- Improved availability of easily used tools.
- Enhanced analyst role while de-emphasizing the programmer role.
- Consolidated life cycle methodology.

The benefits of this approach are substantial. Since the end user is provided with a working model early on, he has the opportunity to clear up misconceptions and because the end user participates in the development process, the end product is more likely to meet with acceptance. The time from conception to initial implementation is shortened and this approach allows smooth transition to improved technologies. Also, selecting an appropriate 4GL for the application can yield cost savings, effectively trading expensive labor for intelligent software.

Prototyping Tools

Our programmer's shop, typical of many other data processing shops, has a large backlog of requests. In addition, his customer has become discouraged with the results of the last system that was implemented. It was months before the system was delivered, and then although it met requirements it really didn't fulfill the customer's needs. Now, another new system is to be developed. The programmer, reading about new tools on the market to increase productivity, is ready to incorporate some of them in his shop. But what do all the terms mean, and which tools will truly improve productivity? A review of some of these tools will help answer these questions.

4GL (4th Generation Language):

This is the name given to any new programming software that will provide a several fold increase productivity over 3rd generation

languages, such as FORTRAN, COBOL, PASCAL, etc. These products range from a product that does one piece of the application to a full application development tool set. A full development tool set usually consists of a screen builder, report writer, database and database interface, and application development language. The screen builder allows the user to quickly create input screens and menus for the application. The database and database interface allows for the storage of data and easy access to that data. The data structures created in the database are easily modifiable for changes and enhancements. Because relational databases are easy to use and easily modifiable, data storage is frequently relational in 4GLs . The report generator provides a tool for quickly producing reports. The language used to build the application can be non-procedural, procedural, or object-oriented. It provides the user with the capability of building either simple or complex applications.

Examples of full tool sets: Oracle, RELATE/3000, Today, Speedware

Code Generators:

These are software which are capable of generating source code. The source code is then available for the application programmer for maintenance, if desired, although more often the generator is used again. Some generators have limitations in the complexity of programming, so it becomes necessary to modify the generated code in complicated applications. However, once modified, the code can no longer be maintained by the code generator. Some code generators have hooks in them to allow execution of programs or routines written in other programming languages.

Examples of code generators: Protos, Artessa

Report Writers:

These are software which are capable of generating reports on specified output devices in formats determined by the package or by programmer specification. Report writers vary in complexity. Different levels of report writers are available ranging from simple ad hoc reports designed by end users, to more difficult reporting formats requiring programmer support. Many report writers designed for end users are menu-driven, where the user can pick from a list of elements, and get the report either on the terminal, or on a printer in batch mode

or while they wait. Many 4GLs have reporting capabilities incorporated in the package. Some of these are Speedware, Powerhouse, and Relate/3000.

Examples of End User Report Writers: Data Express, Inform
Example of Programmer Report Writers: Business Report Writer

Debugging:

This is a feature of a language that helps debug a program. Some 4GLs have built in debugging capabilities which highlight errors as the code is being written. Online debugging features allow developers to modify screen or code without terminating the program. Some, such as Transact, have debugging routines that can be called when problems are encountered during testing.

Examples of 4GLs with built in debugging: Flexible/3000, Relate/3000, Transact

Application Generator:

These are software products which will create the basic program structure automatically, because the functions will normally be similar from one system to another. Filling in values on the menu will define basic transactions, input/output fields and files, and interrelationships between them.

Examples: Insight, Today, Speedware

Screen Builders:

These products contain easy, automated means of creating basic screens from dictionaries/encyclopedias. These are the ideal for rapid prototyping demonstrations for the end user.

Examples of products that have screen builders: Flexible/3000, Today, Powerhouse, Speedware

Database Utilities:

Good database manager utilities allow easy and rapid changes to database design as requirements are obtained from the user during the

prototyping process. Selection of the appropriate database manager is based on the type of access required. Some 4GL products that have full function database managers are dBASE, HPSQL/V and Speedware. Other tools are single function such as Adager or DBGENERAL which modify database structure, or IMACS*LYNX which manipulates data dictionaries. Some tools that enhance performance for database retrieval include OMNIDEX, SUPRTOOL, and SORTMATE

Automated Documentation Tools:

One of the main characteristics of a 4GL is a centralized dictionary for data element and file information. This becomes the core of the system documentation. Some 4GLs have generators that print data in a document format from the dictionary, and from internally maintained files and tables.

Examples of 4GLs with self-documentation feature: Speedware's Documenter, Today

Design Tools:

These tools automate the tasks of system planning, requirements analysis and design. Computer Aided Software Engineering (CASE) products like Excelerator from Index Technology provide front-end design tools for 4GLs. Dictionaries/encyclopedias control the integrity of data element definitions across software packages. Encyclopedias extend the control of the dictionary by containing diagrams of data flow or entity relationships of the system. Software can then be generated straight from the encyclopedia.

Example of utilization of an encyclopedia: SYBASE

Expert Systems:

This is one of the fields in artificial intelligence. This is software designed for a specific task where the knowledge of the expert has been built into the software. There are many software products which will help the knowledge engineer develop an expert system. These range from languages like Prolog and Lisp to expert system shells. With Prolog and Lisp, the developer must start from scratch and build everything that is needed. The shell provides the developer with an environment for ease of developing an expert system.

Examples of Expert System Shells: AION, M.1., Personal Consultant + (PC+), Knowledge Craft

Platforms (Distributed Processing):

To improve the cost effectiveness of the programmer/analyst the use of tools that allow easy porting of software from one machine to another should be used. The computing power on the desk (PCs) needs to be more efficiently used. There are tools that generate software on a PC that can be uploaded to either a mini or large scale system and then run by the end users.

Examples of products executing on multiple platforms: Today, Relate/3000, System Z, Speedware, Oracle, Powerhouse

Terminal emulation software is available for a PC that will emulate a terminal that is connected to a host computer.

Examples: Reflection 1, 2, 3, 4, 5

Prototyping Methodology

Our programmer, excited at the prospect of using all of these new tools, plunges into the task of developing the application. He creates a prototype and demonstrates it to the users. The users are not happy; they suggest several dozen minor and major changes that they need implemented. No problem; the programmer sits down and, using no formal methodology, immediately makes the changes that they have requested. These changes are not exactly what the users wanted either; so they detail further changes. Again, no problem; he puts his wonderful new tools to work and once again makes all of the changes that the users have asked for. The users are a little happier with the application, but there are still several changes that need to be done. This goes on for several iterations. After about the fifth or sixth iteration, the programmer is becoming a little confused. He is no longer sure of the functionality of the system. So many changes have been incorporated that he no longer understands exactly what some parts of this system do. In short, he has created a monster application and has very little documentation to take him through exactly what is going on, and the users are still surrounding him clamoring for yet still more changes. He realizes that in order to regain control over the application,

some kind of prototyping methodology has to be implemented. But what are his choices? After researching many formal approaches that have worked for other people, he decided to concentrate on two methods, Application Development Guidelines (ADG) and Evolutionary Development Methodology (EDM).

ADG was developed in response to customers' needs by the COGNOS company. The steps in ADG's methodology are as follows:

Problem Analysis
Requirements Definition
Data Analysis
Iterative Development
System Completion
Implementation

The purpose of the problem analysis section is to determine the scope of the project, give an overview, state the objectives, define the problem, and to make preliminary plans. The purpose of the requirements definition section is to define requirements, analyze the business functions, define a conceptual solution, design a rough draft of menus and screens, make general plans, and to communicate the approach taken.

The third section is data analysis. The overall objective of this section is to produce a logical definition of the data. There are two different approaches that can be taken to accomplish this. They are entity modeling and data normalization. For prototyping purposes, entity modeling is probably the best way to go.

The fourth section, iterative development, is when the prototyping comes in. Armed with the requirements definition and the data definition, development begins. The data base is created, the system environments are set up, the prototype is written, and user reviews are scheduled. Here the functions are demonstrated. Requested changes are documented, implemented, and tested. The new version is then demonstrated again and the cycle is repeated. This continues on until the users are satisfied with the product or until a predetermined number of iterations have been done. This ushers in the system completion section, where the remaining functions are built, system interfaces and documentation are done, consistency is ensured, the

complete system is tested and is prepared for production. The implementation section is where the system goes into production.

Note: The preceding discussion is based in large part on a presentation given by Marc Praly of COGNOS on July 20, 1987.[2]

The other development methodology EDM is supported by EDM Management Systems. EDM consists of five phases. They are as follows:

Management Study
Technical Review
Feasibility Project
Implementation Project
System Review

The first two phases are done only once, at the beginning of the project. The last three phases are done per iteration.

The purpose of the first phase, the management study phase, is to determine whether or not it is in the customer's best interest to computerize the application. Ideally, this would be done by the customer, since they would have the best handle on what direction they want to take. This whole phase is essentially dedicated to the question, "Does an EDP project look like the way to go?". Outputs from this phase include a definition of goals and objectives, an analysis of the current situation, and a prioritized list of business requirements to be automated.

The purpose of the second phase, the technical review phase, is to determine what the system will cost, and a rough idea of what it will look like. This is done by a systems analyst in response to the output from the management study phase. Outputs from this phase would include a verification of the strategies and assumptions as given in the management study, a conceptual definition of the system, a rough draft of a release strategy, a description of the implementation environment, a rough data analysis and model, and a cost/benefit analysis. At this point the customer's management will either approve or cancel further development.

As said before, these next three phases are done for each iteration. The purpose of the third phase, the feasibility project phase, is to give the customer's management enough information to determine whether or

not to continue on and to implement the proposed changes. Note that if management refuses to authorize these changes, it does not cancel the whole system; only that the proposed changes that were going to be implemented in the current iteration are not to be done. The outputs from the feasibility projects are a definition of exactly what will be done during this iteration and how it will be done, a complete analysis of costs and benefits that pertains to this iteration, a detailed data model of affected areas, a shell prototype system demonstrating the viability of the proposed changes, and the customer's evaluation of the prototype.

The fourth phase is the implementation project. The purpose of this phase is to deliver a working system as defined by the feasibility project and to implement it. Along with this is customer acceptance and training. It is within this phase that the concept of 'Stages' is utilized. Essentially, Stages are used to chop the overall objective of the iteration into bite-sized chunks. The Stages are developed independently, but all must work before the implementation project is completed. The outputs from the implementation project are a final data model for the affected areas, accepted software and documentation, converted data, trained users, and a final review. The final review verifies that the current iteration fits in with both previous and planned iterations.

The last phase is the system review phase. There are two objectives to this phase: to review the success of the iteration and to apply what has been learned to the future. This serves to keep the overall goals of the customer in sight and on track. The outputs from this phase are a review of work done and a release strategy defining future activities.

Note: The bulk of the EDM discussion is based on the document "Evolutionary Development Methodology: An Overview", by Dave Wattling, DWM Computer Systems Ltd. and Douglas P. Kelly, Newton Data Inc.[3]

Potential Concerns

Our programmer was given a list of concerns from his managers to address before being given a green light for assembling his new tool box.

- Has the software vendor thoroughly tested the product before marketing?

- Does the product perform as advertised or are some of the features vaporware?
- Software updates occur at a phenomenal rate. Should purchase be postponed until all the newest features have been incorporated into the product?
- Are his current applications suitable candidates for a 4GL?
- Without experience in the prototyping tools, how does he evaluate which ones are right for his shop?
- Will the new software be compatible with existing software and data base structures? What conversions will be necessary?
- Can our computer system handle the increased overhead usually generated by a 4GL?
- Will our end user be willing to commit adequate time and effort to ensure success of the project?
- Will it be portable to all the platforms currently utilized and how will it mesh with future hardware configuration plans?
- Will performance become an issue if an increase in business places new loads on the system?
- No one in his shop has experience with a 4GL, let alone a whole tool box of new products. New products and a new methodology will move these programmers out of their comfort zone. How long before he can realistically expect training to be complete and the whole shop using the new tools with some proficiency?

In order to answer these and other concerns, our programmer needs to become knowledgeable in a whole new arena of terms and products. This will require manager support, budget, time, interest and hard work. Trade journals can provide background information and a familiarity with the subject. Vendor shows and seminars provide additional information and a review of available products. Considering his shop's requirements, two or three candidates can be selected for indepth investigation. Preliminary questions as to length of time in business, product support, training, etc. need to be addressed. If possible a formal training class for the product could be attended. A talk with someone who has already invested in the product, might yield some candid answers for these concerns. Someone who is already using the product might provide some insight into issues of performance. The evaluation steps need to be a carefully thought out process, involving all the personnel concerned with the project. Only after a complete evaluation can the best possible decision be made for the business.

Data Processing Industry Future

There are a number of general areas within our industry which have shown major advances in the last several years, and we predict will play a significant role in the future. They are Artificial Intelligence, Computer-aided Software Engineering (CASE) technologies, the "Workstation", and Neural Networks. A number of other areas which we will call trends could develop into significant contributors, and will be covered later in this report.

Artificial Intelligence

Artificial Intelligence (AI) is an application technique that more closely resembles the human mind and the way it works. AI uses a number of inventive methods of organizing and accessing data. Several areas of AI have been identified:

- expert systems
- natural language understanding
- perception
- robotics
- machine learning
- planning
- theorem proving
- symbolic mathematics
- game playing

A promising area for the user community, and the most developed at this time is the expert system. The purpose in building an expert system is to replicate expertise and/or to combine expertise for a specific application area. This would result in an increase in productivity by:

- avoiding delays
- distributing expertise to remote sites
- making expertise available to less experienced personnel
- preserving corporate knowledge
- increasing consistency of decisions
- handling routine reasoning and bookkeeping
- leaving an "audit trail".

The medical system MYCIN is a good example of an expert system. It was developed at Stanford in the 1959 - early 70's time frame. Its task is to diagnose and suggest therapy in certain cases of infectious blood diseases. It is a rule-based system paradigm using backward chaining.

Another expert system example is "Prospector" which is a geological prospecting tool. The knowledge is in semantic nets. The savings from using this program alone would have paid for all early AI research.

Several languages (LISP, PROLOG, etc.), have been developed especially for AI to take advantage of the AI concepts, however any language can be used to apply these techniques.

Until recently the AI application community has been restricted by machine size and speed. With the advent of very powerful desk top machines, AI is starting to find its way into almost every PC product. The techniques will become more and more prevalent but will most likely take the form of the application product, and will not be identified as AI technology. Expect to be a user of AI technology and not be aware of it.

CASE Technology

Computer-aided software engineering (known as CASE) has become data processing's newest buzzword. Computer-aided Software Engineering is the automation of software development which focuses on the whole software productivity problem instead of just implementation solutions. CASE technology has been defined as a combination of software tools and methodologies. Software productivity problems are attacked at both ends of the life cycle by automating many analysis and design tasks, and combining them with program implementation and maintenance. CASE products include some grouping of the following tools: diagramming tools for system design, screen painters and report writers for prototyping, dictionaries, data base management systems, and accompanying reporting facilities for storing, reporting, and querying system information, automated consistency checking against system specifications, and code generators to automatically generate executable code from system specifications. In some instances CASE analysis and design tools serve as front end processors to 4GLs.

For now, actual perception of CASE technology seems to be inconsistent and somewhat confusing. And perhaps because the products are so new

to the marketplace, it seems to be an arena filled with vendor hype and vaporware. No full CASE product exists today. Present day CASE products range from the simple generation of bubble charts to sophisticated products which automate requirements analysis through creation of a physical model of a large system, including screens, reports, etc. However, major vendors have brought CASE products to market and future CASE products will automate as much of the analyst's and programmer's job as possible. CASE products may someday add some artificial intelligence, such as an embedded expert system which helps an analyst find his way. CASE seems to be an area which shows real promise as an aid to application development.

Some examples of CASE technology products are:

- The Yourdon Analyst/Design Toolkit from Yourdon, Inc. - provides an integrated data dictionary for error and consistency checking throughout the analysis and design phases.

- ProKit*Workbench from McDonnell Douglas - fully automates the application of structure techniques of planning, analysis and design phases.

- The Design Machine from Ken Orr and Associate, Inc - generates the deliverables of system requirements and analysis.

- DESIGNAID from Nastec Corporation - automates structured analysis/design techniques.

Most of the CASE tools on the market place today support various phases of the Life Cycle Methodology. CASE tools to support more rapid development methodologies are expected to emerge, but we are not sure of the form or timing of these products.

Workstation.

We define a workstation as a microcomputer on an employee's desk which has at least the capability to do word processing and spreadsheet processing and is connected into a "Workgroup" which allows access to other computers in the network (could be other PCs, or minis, or mainframes).

During the last few years more and more employees are getting microcomputers on their desks. In many cases these are replacing dumb terminals and their prime usage is to access another computer (mini or large scale). This computer power on the desk opens up new methods of doing normal office operations. Microcomputer applications are increasing in usage and importance within the company. The most common usage today is in word processors and spreadsheet processing.

As more powerful microcomputers become available, and as we become more proficient in using this local computer power, it will play an increasing role in the DP industry. The AI techniques discussed above and the CASE tools will have a prime focus on the micros and not on the more traditional minis or large mainframes. These machines will be included in the workgroup which may include a complete range of processors networked together. The possibilities of these networks are just now beginning to be realized. Electronic mail is finding its way into the work stations as well as some object-oriented programming techniques.

Neural Networks.

This is the general area of applying biological memory techniques to networks of silicon memories. This provides a better replica of the human brain than we have seen before. Recent breakthroughs in the technology have made this technique very promising. It is expected to provide considerable speed as well as vast memory storage with rapid retrieval. The technology is still in the research labs, but looks to hold such promise that it is mentioned in this report.

Trends

What does the future hold as seen by the prototyping team? We see this as a very exciting and rapidly advancing time for the data processing industry. The items mentioned are the best guess at this time of what's coming in the future. We look at this section as being very speculative. We will address these trends in several areas, future hardware, future software, where is DP going, and the end user.

Future Hardware

There is a trend to place large amounts of computing power on the desktop. This will have a major impact on large scale and mini

computer usage as the applications migrate to the desk top computers. Jean-Louis Gassee, Apple Computer's Senior Vice President of Research and Development, has a dream of placing a 200 mip PC on every desk .[4] He is not sure that he will see it in his lifetime, but he is sure it will happen. Does this mean that large scale and mini computers are going the way of the dinosaurs? It may mean that machine size and speed will no longer be a factor, but how the machine(s) are used will be.

Networking is beginning to mature. The hardware is available to virtually connect any computer to any other computer. We expect this function of our business to continue maturing and will become a major cost factor in the future. This connectivity will make electronic mail a true reality. The desk top machine will have the phone built in and have FAX capability.

Disk and memory continue to become less expensive. As this trend continues all the known restrictions on memory and storage space will disappear. This, along with networking and larger desk processors will make distributed processing a household word, not the dream it is today. Expect to see desk top machines with 16MB memory and many gigabytes of storage commonplace within the next few years.

Hardware is being placed in office environments instead of a computer machine room with special environment requirements. As this trend continues expect to see the operations tasks normally delegated to data center personnel turned over to the user community as is the hardware itself.

Hardware technology has been progressing much faster than the software industry can keep up with it, and the user and DP communities cannot keep up with the software changes which are much slower than the hardware changes. As users and DP begin to use these new hardware technologies expect major changes in the way of doing business. Micros and Minis are blending into one machine which looks more like a micro than it does a mini of the 1970s.

Future Software

Expect CASE tools to eliminate the job of the system analysts as we know them today. Coding will be a thing of the past. The CASE tools will allow an analyst working with the user to build and maintain a system with much less effort than is currently being spent. Code

generators will be built into products and not even discussed. Since labor is more expensive than hardware, functionality will be the focus, not efficiency of hardware resources. The CASE tools may allow the software designers to keep up with the hardware designers.

Look for an increase of user friendly software. Many of the software tasks now done by the system analyst will be taken over by the end users. End user reporting is the most common item in this area today although expect it to extend into system design and maintenance.

Expect to see the life of a software system to be shortened considerably. This is caused by a number of factors:

- New tools make the initial investment much smaller.
- Rewriting the software may be more cost effective than maintaining it.
- Rewriting the system to make it run on new inexpensive hardware may be cost effective.

AI techniques will be used in all software. The CASE tools will generate software which begins to think and adapt itself.

Object-oriented Programming is just emerging. Expect to see more packages such as Hypercard on the Mac which use Hypertext technologies, and allow end users to graphically build their own applications.

DP Trends In Business Applications

Systems will be able to be designed very rapidly. Does this mean that full time DP Analysts are no longer needed? Some people think so, so expect to see a migration of DP expertise drifting to the user communities in part-time DP positions. Others feel that as it becomes easier and less expensive to generate computer systems, the user community will demand many more systems than they are now doing, thereby increasing the need for DP analysts. In either case, we all agree that the way DP is doing business today is going to change.

The DP role will become more specialized. The DP analyst will determine network handling and provide consultation on the project

design and development. The end users will do nearly all of their own reporting requirements, and most of their machine operations and simple changes to the system.

What does this mean to the analyst of today and their DP organizations? Should they be looking for a job in the user community? It appears that programmers may need to either move to the user community or to develop their design skills. Analysts who will move into either the network areas or the consultation areas must become more skilled on the tools available and how to apply these tools to make them more productive. Communication skills will become more important both written and verbal. During a transition period old systems will need to be maintained and old data centers need to be staffed until both of those areas can be phased out to the new methodologies. Plans need to be made on how to make this transition most efficiently.

Data centers as we know them today may only be needed in extremely large applications. Most of the mini data centers will be replaced by user installed and run machines in their own areas.

What should the DP analyst of today be doing? We suggest the following:

- Keep abreast of what is happening and react in concurrence with your personal goals in the Company/Industry.

- Get as much training as possible in new technologies.

- Work on your communication skills.

- Learn as much as possible about your customer's way of doing business. This knowledge can give you an edge.

End User Trends In Business Applications

As the generation of applications and reports become easier, it is possible for the non-DP professionals to develop their own applications and reports. Many users in our shops are currently using ad-hoc report writers to do their own reports. In the Apple community, many users are developing their own applications and reports using Hypercard.

Training vs Education

We do a very good job of providing training on specific tools but do a very poor job of educating our DP community or our end users of the trends identified here. More emphasis must be placed on education to allow all computer users to more fully use the computer potential in their normal daily jobs. The tools must be made easier and more friendly so they do not get in the way of getting the job done. (i.e. We have had enough of stick shifts, we now need automatics.)

In Summary

Competition to increase productivity will force change. The utilization of new technologies and methodologies to achieve improvement in software application development becomes all important. Preparation and education are the keys. The better we understand what the new tools and hardware can do for us, the better we can make them work for us. Management needs to be committed to an early selection and acquisition of new products and to the training of their programmer analysts not only in the use of these tools, but also in the effective use of a controlled prototyping methodology. Establishment of a closer working relationship with the end user becomes a necessity. Application Prototyping dictates the early and continual involvement of the end user in the development process. Development time will be shortened and the resulting software will have a better chance of user acceptance.

---

[1] Ooops, now we've done it. Well, for ease of writing as well as reading, all gender references in this paper will be masculine. The reader is politely asked to mentally switch gender gears as he/she finds necessary.

[2] Marc Praly, Sales Representative, COGNOS Corporation
Training presentation at Boeing July 20, 1987

[3] Dave Wattling, President of DMW Computer Systems Ltd.
EVOLUTIONARY DEVELOPMENT METHODOLOGY: An Overview.
May 1986

[4] Statement from the Keynote Presentation at MACWORLD Expo in Boston on August 13, 1988.

PRODUCTS LIST

Adager	registered trademark of Adager De Guatemala, S.A.
AION	registered trademark of AION Corporation
Artessa	product of RAET Software Products
Business Report Writer	registered trademark of Hewlett-Packard Co.
Data Express	trademark of Imacs Systems Corporation
dBASE	registered trademark of Ashton-Tate Corporation
DBGENERAL	product of Bradmark Computer Systems Inc.
DESIGNAID	registered trademark of Nastec Corporation
DESIGN MACHINE	product of Ken Orr and Associate, Inc.
Evolutionary Development Methodology	registered trademark of EDM Management Systems Inc.
Excelerator	registered trademark of Index Technology Corporation
Flexible/3000	product of Sages-America
HPSQL/V	registered trademark of Hewlett-Packard Co.
Hypercard	registered trademark of Apple Computer, Inc.
IMACS*LYNX	registered trademark of Imacs Systems Corporation
Inform/3000	registered trademark of Hewlett-Parkard Co.
Insight	product of Computing Capabilities Corporation
Knowledge Craft	registered trademark of Carnegie Group Inc.
M.1.	product of Teknowledge Inc.
OMNIDEX	product of Dynamic Information System Corporation
ORACLE	registered trademark of Oracle Corporation
Personal Consultant +	registered trademark of Texas Instruments Incorporated
Powerhouse	registered trademark of Cognos Incorporated
PROKIT*WORKBENCH	registered trademark of McDonnell Douglas Information Systems
Protos	registered trademark of Protos Software Company
Reflection	registered trademark of Walker Richer & Quinn, Inc.
RELATE/3000	registered trademark of Computer Representative, Inc.
SORTMATE	product of Pacific Coast Building Products, Inc.
Speedware	product of Infocentre
SUPRTOOL	product of Robelle Consulting Ltd.
SYBASE	registered trademark of Sybase, Inc.
System Z	product of Zortec, Inc.
TODAY	product of bbj Computer Services, Inc.
Transact 3000	product of Hewlett-Parkard Co.
YOURDON ANALYST/DESIGN TOOLKIT	product of Yourdon, Inc.

Managing System Performance

By Laurie Facer
FACER INFORMATION DESIGN
PO Box 270 Epping, Australia 2121
Phone: 011 61 2 484 3979 Fax: 011 61 2 484 5709

There comes a point in every System Manager's lifetime when he feels that he should "really do something about system performance". However, the reality of the situation is that controlling system performance is more easily said than done.

Good system performance on a given CPU is not achieved by the implementation of one simple idea. In reality, the system performance experienced on a machine at any one time is the accumulation of many decisions made over time and often made in isolation to each other.

The multiplicity of factors involved in controlling system performance, does not diminish the fact that system performance is crucial to the success of any DP installation and ultimately the organization itself.

Why A Strategy?

The starting point in tackling system performance is the recognition that system performance is not the result of any one decision. Rather it evolves over time and is the result of many decisions.

The operating environment of the HP3000 is extremely dynamic. There are many variables that make up the total environment and, indeed, that environment changes continuously. The environment is dynamic in two ways. Firstly, given a particular environment structure, the interaction of processes within that structure changes constantly and secondly, the environment structure itself changes over time.

Let me give an example. A machine that has exactly the same file structure and data on two days may behave differently on each day. The reasons for this are manifold - different pattern in logon times, two programs run together on one day but not the next, somebody decides to do a KSAM generic search, etc.

The structural environment has remained unchanged, but performance has varied markedly.

Of course the structure of the environment itself is constantly changing with the addition of new hardware, the introduction of new systems, converting an IMAGE database system to an OMNIDEX database system, etc.

Another complexity in system performance is the inability to find many absolute rules that will always ensure good performance. One rule may work beautifully for one installation, but have horrific

results for another. These rules even change on the same machine, depending on what is happening at the time.

For example, very large blocking factors may be fantastic for overnight batch processing, but kill the system during the day for data entry and online reporting.

To further add to our problems, system performance is not obtained by just concentrating on one variable. System performance is controlling many variables.

Obtaining the most from your HP3000 is a balancing act - everything must be kept in balance so that bottlenecks are not created. If bottlenecks do occur, then waiting occurs, and poor responsiveness results.

The number of variables to determine that balance are enormous - files balanced across discs, on-line processing versus batch processing, memory size, code segmentation, blocking factors - and the list never seems to end. Just to think about where to start gives one a headache. And just after you have it all sorted out, you go to a user group meeting and somebody tells you something that totally contradicts the thing you just spent three weeks in implementing.

So, when tackling system tuning we are faced with three basic problems:

a) Dynamic Environment - moving target syndrome.

b) Absence of many absolutes - the theory of system performance relativity "Everything is relative to everything else".

c) Unlimited factors - how long is a piece of string syndrome.

Given these problems, obtaining system performance is not just a matter of twigging a few bits - it is a continuous operation that needs constant management. If the system manager is to obtain control over system performance, he needs to first develop and then implement a clearly defined strategy.

The Strategy - A Plan Of Action

In developing a System Performance Strategy, three tasks need to be performed:

a) System Performance Goals set.

b) Action Plans detailed.

c) Resource requirements defined.

I need to emphasise at this point that system performance is a project just like any other software project. The size of that project depends on how far you want to go in controlling system performance.

There are no "go fast" buttons on the HP3000. As stated earlier, system peformance is the final result of many decisions. The better control you have over the decision making process, the better your system performance will be.

Unfortunately, many system managers are not in the position to exercise total control over the system. This fact needs to be recognised in the strategy and goals set accordingly.

System Performance Goals

Before goals can be set a definition of system performance must be formulated. The definition that I propose comes from Neville Silverman, the author of the system performance product, CIA.

Neville proposes the following "System Tuning Definition":

At the process level:

Minimization of CPU usage.

Minimization of Disc I/O usage.

At the Global Level:

Minimization of on-line Process response time

Maximization of Global Process CPU usage.

Minimization of Queue lengths on each Disc drive.

Minimization of Memory Manager activity.

Minimization of Dispatcher activity.

At the Process level we are looking to minimize the resources required by each process to perform their allocated function. This implies control over decisions made at the programming and system design level.

At the Global level we are looking to maximize the "effective" use of resources and minimise the overhead in using those resources. We are also looking to minimize bottlenecks such as queueing on disc drives.

You may wish to construct your own definition of system performance, but I believe the above definition to be a good starting point.

Having obtained a definition of system performance, the next step is to construct a series of goals. These goals should be set in such a way that their achievement can be measured. To be able to this you may require one or more system performance measurement tools. Preferably, the purchase decision for that tool will be made after you have defined your system performance strategy.

The goals that you set should be tailored to your organizations particular requirements. I will propose four simple goals for a fictitious installation.

Goal I

Session Response Time is to average less than three seconds between 9 am and 5 pm Monday through Friday.

Goal II

Overnight production will complete by 6 am Monday through Friday and 5 pm Sunday on weekends.

Goal III

"A" priority reports will be completed one hour after requested.

"B" priority reports will be completed four hours after requested.

"C" priority reports will be ready for distribution 6 am next day.

Goal IV

Client Account Enquiry - Will take less than 4 seconds to enquire on a client's account at any time.

Bank Reconcilliation - Will be completed at 4 pm Monday through Friday.

Unmatched Deposits Report - Will always complete within 5 minutes of start.

My first three goals are general goals representing the processing standards that are expected from my installation. The fourth goal is specific to the requirements expected from specific applications. These are the goals by which I will rate my system performance.

Developing A Strategy

Once the goals have been defined, a strategy for attaining those goals needs to be developed. The strategy that I would recommend has five components:

1) A System Performance Monitoring System that reports progress in goal attainment - Goal Reporting.

2) A System Performance Monitoring System that provides enough data on system activity to allow the capture of system performance culprits - Activity Reporting.

3) The definition of areas of system performance that most urgently need attention.

4) The creation of projects to tackle the renegade system performance areas.

5) Performance review procedures for new programs and systems.

The Goal Reporting system highlights the system's performance against the stated goals. Only if there is a deviation from these goals, need any action be taken.

If action does need to be taken, activity reports must be available to allow detailed examination of system activity. These reports should be able to be turned on or off depending on the need for troubleshooting.

Once you have gained control over system performance, the need for projects to examine system performance problems will be greatly diminished. If, however, you are faced with a badly performing system, you will need to tackle one thing at a time and progressively gain control over performance.

To maintain control over system performance, a set of performance review procedures is essential. You have just spent a lot of time and, perhaps, money to get better system performance. The procedures you implemented to gain that performance need to be propagated in new programs and system designs.

Goal Reporting

For Goal Reporting, I will concentrate on just one of our objectives - Session Response times. This goal is generic to a well performing system and is the most applicable to the vast majority of HP3000 installations.

Goal I stated that our system is required to have a Session Response time that averages less than three seconds between 9 am and 5 pm Monday through Friday. To determine if this goal is being fulfilled, we need to be able to measure Session Response times. The problem with this is defining what a response time is.

Superficially, response time is the time it takes to receive a response after hitting a key to send data. But in reality, life is a lot more complicated than that.

For example, if an HP3000 was only running one process, the response time using 9600 Baud would be faster than a response time using 1200 Baud. Sometimes it is hard to know what is a request for something to be done and when the task is complete. V/3000 sends characters down during machine "think" time - these should not be mistaken for user data transmission.

A further example of difficulty in measuring response times is highlighted by HPWORD and online QUERY reports. HPWORD is continously transmitting and receiving characters. We need less than three seconds response time for that. However, if someone requests an online serial read from QUERY, it is a bit unrealistic to expect a three second response time for that process.

Response time will vary widely by the minute, as the exact functions that the machine will be performing at any one time is totally unpredictable.

There are products available that will measure device response times, eg, TMS from Orbit, and these are very handy for measuring responses of particular devices.

Another approach that can be taken is the one similar to that taken by CIA's Session Reponse Time Report.

This report shows, for sessions divided into "C" and "D" queues, the average response times for specified time intervals for all active session processes. Just as importantly, it shows the wait times that processes experienced for system resources. The response time provided by this report is the total wait time divided by the number of Terminal Faults.

Recognising that not all processes can be treated equally, a third column is provided showing response times for individual processes nominated by the system manager.

This report has three major advantages:

a) Individual processes can be excluded from overall figures and examined separately.

b) The response time is a good approximation of wait times for processes and ignores transmission rates.

c) The wait times highlight the areas causing most delays.

This, or a similar report, can provide our benchmark for determining how Goal I is being achieved.

As stated before, system response times are the result of the sum of the individual parts. Having established the benchmark report for overall performance, the next step is to monitor the individual parts. This is done through the activity reports.

Activity Reports

It is convenient to group system activities under three general headings:

Global Activity

Disc Activity

Process Activity

Global Activity

Global activity can be divided into three major parts:

CPU Activity

Memory Manager Activity

Dispatcher Activity

According to our System Tuning definition, we want to maximize efficient CPU utilisation, minimize Memory Manager activity and minimize Dispatcher activity.

The CPU is where the work gets done. The objective should be to maximize the time CPU spends on working for processes and minimize the time spent on performing other tasks.

For convenience, most performance analysis products divide the work performed by CPU under the following or similar headings:

B - Busy

P - Pending I/O

C - Caching

M - Memory Manager

I - Idle

V - Virtual Memory

O - Dispatcher and overhead

A machine under stress will see low Busy percentages and high percentages in other areas. To gauge bottlenecks in a system, the CPU busy state is a good place to start. High percentage levels for each state defined above points to the following problems:

B - Processes are obtaining a high work rate from CPU. The higher percentage time spent in this state the better.

P - CPU is pausing a lot and wating for I/O. If you have a lot of processes running and you are obtaining high "P" percentages, then your machine is incurring I/O bottlenecks.

C - Caching percentages will be high if the system is spending a lot of time managing disc caching. If you have constantly high "C" percentage and low "P" percentages with many processes running, you may need to rethink your disc caching strategy.

M - When memory is scarce, Memory Manager works overtime. High memory activity can also point to bad program segmentation.

V - Similary, when "V" is high, this indicates a lack of memory, as transfers to and from virtual memory are made necessary.

O - Overhead represents the CPU utilisation of Dispatcher. Dispatcher controls entry into the CPU. If it is constantly high, your machine is under stress.

If we look at CPU busy states, along with Dispatcher activity, we have a very powerful indicator as to how things are performing.

The CIA product represents Dispatcher activity through "launch" rates. A launch rate is the number of times that Dispatcher launches a process into CPU. If Dispatcher has high launch rates, it means that the available resources are having a hard time keeping up with processing demands.

Earlier, I stated the theory of system performance relativity, that is, everything is dependent on everything else. The above indicators are a classic example of the application of that theory. You cannot run a system performance analysis tool, see a high memory percentage and then assume you have memory problems. You must look at the CPU busy states and the launch rates relative to each other and your benchmark report - the Session Response Time report. To do this, you must be able to cross reference by time of day and be able to look at the day in its entirety.

If response times are high, launch rates high and the busy state of CPU is low, obviously you have a machine that cannot cope with processing demands.

If, on the other hand, you have low session response times and your launch rates are low with CPU spending most of its time in the "busy" state, your machine is coping very well.

Disc Activity

In monitoring disc activity, there are two areas that need attention:

Disc queue lengths

Relative work rates of each disc

Disc queue lengths point to the inability of a disc to cope with transaction volumes. If queues are continuously sitting at a length of six or greater, then that disc is under stress. If the system disc is continuously sitting at queue lengths of six or greater, then the whole system is under stress.

It is absolutely critical that the queue lengths on system disc be kept low. MPE keeps its tables and directory on system disc. If you open a file on disc 2, MPE has to go to the system disc to look up the directory before going to disc 2. If there is excessive queueing on the system disc, then access cannot take place on disc 2.

It is also important to maintain balance across discs. Those discs with extensive queues could have those queues reduced by shifting active files to the discs with lower or no queues.

It is imperative then that you have an activity report showing the queue lengths on each disc over periods of time. When using this report trends can be highlighted and corrective action taken.

Process Activity

To obtain total control over system performance, the system manager needs as much information on process activity as possible.

The process information required includes when the processes ran and the resources they required during their execution time. This detailed process information can then be crossed checked against the global and disc reports to give a full picture of machine activity and how well the machine coped with that activity.

Defining Problem Areas and Projects

Once the system performance monitoring system is in place, system performance problem areas can be defined and projects to tackle those areas established.

I stated earlier that system performance rules are rarely absolute. Having said that, I will outline several reasons for poor system performance and give the symptoms to look for. If you have a poorly performing machine, you need to start somewhere, and what follows is a list of some of the areas to start looking. I will give these examples in the form of cases.

Case I - Too many processes running simultaneously.

This is a relative easy one to pick. When there are too many processes screaming for attention, Dispatcher works overtime. Firstly, you will notice very high launch rates and associated with this will be very high pre-emption rates. This is the result of high levels of faulting (loss of access) in the CPU as MPE tries to resolve the imbalance between available resources and process requirements.

You will also notice that the CPU busy state will be low and the other CPU states (particularly overhead) will be high.

Obviously session response times will be poor. You will also notice that CPU faulting rates will be high and in particular preempt rates will be very high.

To resolve this situation, besides making all processes more efficient, you have two major options.

The first option is to upgrade the CPU. Leave things as they are and upgrade. This is an expensive option and is great provided that the system load does not keep growing. Sooner or later you are going to either run out of CPU upgrades or money.

The second option is to spread the work load. If you have heavy workloads during the day, but light workloads during the evening, you have the ideal opportunity to ease the daytime workloads by running at night. I strongly urge all installations to look at this option. The night hours are often full of unused processing power.

Workloads can also be spread by ensuring that the job limit is not high. If there are more than say three jobs running concurrently with sessions, then this will cause stress. Remember that jobs do not wait for terminal faults. They will take all the resources they can until they are impeded or pre-empted by a higher priority process. Pre-empts cause dispatcher a lot of work and will impact session response times.

Case II - Lack of Memory

Under these circumstances, Memory Manager works overtime. You will see the CPU spending a lot of time on memory, virtual memory will be high, and, if you have caching, CPU time spent on caching will also be high. Again the busy state will be low.

Also, pay attention to memory allocation rates and the memory cycle time.

Memory allocation is the rate per second that attempts are made to find space in Memory and to make an absent segment (i.e. in Virtual Memory on disc) present. The Memory Cycle rate is the rate per second that Memory Manager went completely through Memory. Each cycle will endeavor to force out Memory segments that should really remain in Memory.

The simplest and most effective solution to this problem is to increase your memory. However, the problem can be eased by ensuring a good balance of workloads and if you have disc caching on, then turn it off (either for all discs or for selective discs). The dynamics of caching are such that the only effective way to control disc caching is to have it managed it automatically. The product CIA provides this facility.

Case III - I/O Bound System.

I/O is potentially your worst enemy for system performance. It is the slowest part of the machine and is all important for any processing.

An I/O bound system will show the CPU with a high percentage of time waiting for "Pending" I/O.

Most noticeably, you will see a high I/O rate for disc drives and extensive queueing on each disc.

The remedies for I/O bottlenecks seem to be endless. Make sure that the workloads on each disc are well balanced by moving active files across discs. Ensure that there are no active files on the system disc. Review blocking factors on files. Review file structures for efficient access to records and ensure that programs are efficiently using disc accesses.

There is enough to be done in the I/O area to ensure several projects. It is also important to constantly review I/O variables in new systems and programs.

Case IV - Some Processes Are Very Inefficient

Weeding out the inefficient processes is a very important part of system performance. You need to be able to identify the culprits so that they can be corrected or at least recognised and rescheduled for less busy times of the day.

If we go back to the System Tuning Definition, there are two things that we require processes to do:

Minimize CPU usage

Minimize Disc I/O

The method I suggest for tracking down the most harmful processes is as follows:

Look at the Session Response Time Report for the worst periods of response time. If you are using a report such as CIA's Process Level Statistics Report, list all the programs that were running during the bad response time period.

Highlight those procedures that have a high CPU second time in relation to run minutes and that have a large number of sectors moved. These are the processes that are chewing up CPU and I/O.

Each program then needs to be individually evaluated to determine if:

1) They should be running at these times of day

2) If they can be made more efficient

An alternative approach is to look at the Process Level Statistics Report for the entire day and highlight those processes that use the most CPU and shift the most sectors of disc. Making these processes more efficient could help system performance overall, not just at particular times of day.

Defining Resource Requirements

If you have a badly performing system, it is going to cost money to correct it. Either a hardware upgrade will be required, or resources will need to be allocated to review existing programs and systems.

Hardware upgrades are solutions that yield quick results. Unfortunately, they may not be the best solution. Software evaluation is the best long term solution, but it takes time.

If you undertake software review and evaluation you will ensure maximum utilisation of hardware resources and have greater control over your system. The potential benefits of this approach are substantial, but you must be prepared to spend time and effort.

There are four prerequisites for undertaking a successful system performance review of software.

Firstly, like all successful software projects, you must have management on side. You will be making recommendations that may cause disruption to current procedures and require investment in new systems.

For example, the installation that is suffering from poor process scheduling will need to reschedule resource hungry processes to "out-of-hours" processing. This means shifting those reports that do endless serial reads during the day to processing overnight. It may be hard to convince some users that they cannot expect to get their reports until the next day.

That same installation may need to invest in a good job scheduling and dispatching system and a product like Omnidex to eradicate serial reads. To make such an investment, management needs to be onside and to understand the goals being sought.

The second prerequisite is technical knowledge. It is no good trying to make a system perform better if nobody has the technical competence to do so. But if you are a system manager of a small shop who has little technical expertise, do not dispair. Technical expertise can be obtained from several sources.

If you have a good system performance monitoring system in place and a strategy for tackling system performance, you can harness the knowledge from others and obtain knowledge from experimentation. The whole point of a system performance monitoring system is to reveal what the system is doing. This information obtained on your machine can be utilised in discussions with third party vendors, HP System Engineers, and consultants. You are no longer at the mercy of a lot of "maybe it is because ofs" and "it probably would be better ifs". You can provide figures that can be evaluated and you can experiment with suggestions and measure the results. You have an ideal learning environment.

The third prerequisite is to have the ability to change your operational environment. Many system managers feel that because they are dependent on a third party supplier and cannot write programs themselves, that there is little they can do about the performance of the software. This is not the case. There are many changes that can be made to improve system performance that do not require one line of code to be changed. This is particularly so in the area of I/O. Master sets in databases can be given more efficient capacities, detail datasets can be better reorganized, blocking factors on files can be made more efficient, files can be better spread over discs, and the list goes on.

There is a lot of literature on ways to improve system performance. Set yourself a project of improving I/O and armed with this system performance material and your system performance monitoring system, clean up existing I/O bottlenecks.

Also, tell your third party supplier that you have performance problems and then show them the reports from your system performance monitoring system - it might help them to "concentrate their minds" if they know that someone is watching.

The last prerequisite required to successfully review software for system performance is the ability to be able to establish software review and implementation procedures. Obtaining good system performance is a constant battle. There are operational procedures that must be performed on a regular basis, programmers and systems analysts must be aware of the repercussions of their programs and system designs, and users must be able to communicate expected increases in transactions.

The system manager's work is never done.

System Performance Monitoring System

So far I have outlined the elements in developing a system performance strategy. An essential part of that strategy is putting a system performance monitoring system into place.

As well as providing the information necessary to develop and implement a system performance strategy, the system performance monitoring system has two additional major functions:

1) Communicate system performance progress to management

2) Facilitate capacity planning

Communicating System Performance

System managers invariably find that the two hardest things to sell to management are system upgrades and utility software.

If there is a system performance strategy in place and the system performance monitoring system is reporting goal achievement, then management will be better able to relate to the system managers requests.

The requests are no longer - "we must buy this upgrade product because we have performance problems". Rather the request can be presented as a well-defined need and solution.

Management can now be approached with the proposition - "As can be seen by the CPU graphs and the Disc Analysis graphs, we need to purchase this software to overcome an I/O bottleneck. This should reduce our current response times from 5 seconds during the day to less than 3 seconds".

Management will be much more receptive to a request that shows potential savings.

Capacity Planning

Capacity planning is simply an extension of system performance monitoring.

When undertaking capacity planning, there are two things that you need to know:

1) What will be the growth rates of my current systems

2) What new systems will I need to carry

In looking at system growth rates it is essential to weed out the important from the unimportant. Using the process analysis reports, it is easy to construct a report that shows the most resource hungry systems on your machine.

Once you have isolated those processes, the next step is to see how well you are performing against your system performance goals. This gives you a tolerance factor. You can then make assumptions about growth rates for each system and see the possible extra workloads that will be placed on the system.

For example, if the accounts system takes up 50% of processing requirements and this system were to increase by 20%, then if we are sitting on the response time goal, we will be pushed over our goal targets.

On the other hand, if we are well below our response time goal, and the system accounted for 20% of the system resources, then a 20% growth rate can be easily accomodated.

For new systems, it is best if you can approximate that system to existing systems, and then extrapolate on the existing system.

Capacity planning is a very imprecise science, but at least we now have the performance information that will allow us to make educated guesses.

In Summary

Developing a system performance strategy is essential in gaining control over an installation. If the performance of a machine cannot be defined then basic decisions on hardware upgrades, software purchases, etc, just become a shot in the dark.

There are three elements to developing a system performance strategy:

1) Putting in place a System Performance Monitoring System

2) Formulation of system performance goals

3) Implementation of system performance projects

The first is required to measure progress and obtain information for sound decision making.

The second is required to obtain a clear communication of system performance objectives.

The third is required to enable system performance to be tackled in a rational and rewarding manner over time.

IMPLEMENTING SALES FORCE AUTOMATION SYSTEMS

BY MARK P. SHIRMAN
INNOVATIVE INFORMATION SYSTEMS (IISI)
63 Nahatan Street
Norwood, MA 02062
(617) 769-7511

Introduction

Automated systems have become commonplace in almost every industry providing solutions in a variety of different application areas. There are scores of potential software alternatives for Accounts Receivable, General Ledger, Order Processing, and Inventory Control, but, the whole area of automating the Sales Force is a concept that is just coming into its own. For many years, salespeople have had to rely upon detailed recordkeeping and dogged determination to insure that sales prospects did not fall between the cracks. Additionally, management had very little way of accurately measuring the performance of marketing campaigns, telemarketing efforts, and their sales force as a whole. This paper will provide an introduction into the world of the fully automated sales force. We will discuss some basic philosophies as well as explore some of the features and functions that should be present in a sales lead management solution.

WHY AUTOMATE?

There are literally dozens of reasons why a company should consider automating its sales force. Some of the major benefits are as follows:

- Provide a tool for the sales reps so that leads are appropriately and completely followed up
- Provide a tool for management to gain better control over not only the sales side of the business, but the entire business operation
- Provides a beneficial Return-on-Investment
- System can provide an interface to the other automated applications in place
- Allows more comprehensive and detailed marketing analysis to be performed
- Provides a method to standardize the approach to sales so that personnel turnover can be more easily managed

- Provide an easy way to import or handle mailing lists from outside sources as well as export information to other systems or companies

The first of these benefits, a tool for the sales reps, can easily be put into perspective through the following graphic:

WHY DO CUSTOMERS STOP BUYING?

1% Die
3% Move Away
5% Buy from a Friend
9% Buy from Competition
14% Product Dissatisfaction
68% Because of no contact or Indifference or Attitude of Sales Rep (Source: Edward Leader, Professional Sales Seminar 1981)

These are some pretty telling statistics. In short, the more leads that a rep can follow up or touch on the more business they will generate. However, it can be an extremely difficult task to reach all the potential leads in an orderly manner, especially if the sales rep is already overburdened. An automated system can provide a rep with the tools to develop an organized approach to lead tracking.

Most good sales lead tracking software can generate tickler or action work files for reps to work off of. These files can be generated based upon followup dates, client interest, geographic location, product interest, and so on. This provides the rep with a more focused view of what they have to do for that day or week. The lists can be used to fuel either telemarketing operations or for personalized sales calls. The average cost for a sales rep to make a call in the U.S. is about $300. Based upon this fact alone, it would appear wise to be able to selectively followup and track those leads that are of the highest potential.

How does management gauge the success of their sales force? The obvious answer is through the numbers of orders placed, however, what they don't know is what percentage of business fell through the cracks or was generated from various marketing campaigns. Additionally, due to the generally independent nature of sales reps, it can be difficult to standardize the approach to lead tracing. This can become a serious problem when an individual leaves their job. Quite often valuable time and energy is lost as people try to figure out what the rep has left behind, if they did leave something behind.

An automated sales lead tracking package can assist organizations with many of the problems mentioned above. Most packages will be able to track the origin of a lead. This can be extremely helpful, especially in evaluating the success of a particular marketing/advertising campaign. The typical example would be an ad that generates a lot of "bingo card" type leads, or requests for product information. An automated system can help determine what percentage of those leads actually led to a sale as well as what the overall potential of these were. Additionally, lead tracking software has the ability to handle a large influx of leads that could be a result of a major campaign.

Management should view sales force automation as a tool, in the same manner the sales rep does. By having access to all the leads for an organization, management can accurately create forecasts, measure rep performance, and standardize its approach to lead followup. The key here is the increased level of control automating the sales force can bring management. In this way, it is no different than automating a manufacturing facility. Organizations often pursue this path in an effort to control costs and analyze more data and it is no different with sales lead tracking systems.

Management should also be impressed with the return-on-investment figures that usually correspond to automating the lead tracking functions. The following pages represent some models that can be used to evaluate the ROI figures. We will start small and think about a company with annual sales of one million dollars and a software package that costs $14,400. You can see that even at a 5% increase in business the ROI figures are amazing. That 5% figure is much lower than many statistical surveys have shown. In fact, Sales and Marketing magazine found productivity increases averaged 43% (Summer 1987) with the introduction of an automated sales lead tracking system. The ROI figures coupled with the benefits mentioned before make a pretty compelling argument in favor of automating the sales force.

Philosophies and Approach

There are two schools of thought when it comes to achieving successful sales force automation. The first school believes in a totally decentralized approach typically revolving around microcomputers. The key here is to provide the sales force with information and tools so that they can manage their own sales calls and followups. This may be the most flexible solution, but may lack overall cohesiveness and consistency. Systems that are almost entirely PC based may only be used by the most aggressive sales personnel, since it may become too onerous for the average rep to see any benefit from the system.

The second school is based upon a centralized data base theory, with a HP3000 in the center and terminals or PC's connected to the main CPU. A centralized IMAGE data base allows sharing of information and files much easier than with PC's. One of the goals of sales force automation is that of management control. By having a central receptacle for data, this is more easily achieved. The central data base concept can also have implications as to how the system can interface with other automated applications. MIS should not view the sales lead tracking data as simply an "island of information". It is possible to feed order entry as well as accounting systems, and manufacturing systems the information generated by an automated sales tool. Conversely, the sales software should be able to "import" information from other data base applications. This entire concept will allow easier generation of management reports and development of inquiry subsystems.

My preference is a hybrid type of system. This approach would have a HP3000 at the center with a combination of PC's and terminals hanging off it. The PC's would have the ability to act as standalone systems, but would be able to communicate to the central system for both sending and receiving information. In this scenario, the package should have the ability to "export" information to other processing options, ie. another HP3000, Vectra or Portable. The concept can be thought of as distributed processing with an interface to the host. Let's consider an example. A sales rep is in the field. He/she downloads a list of leads from the corporate office to a PC. The leads are managed, tracked, changed, and more are added. At the end of the day, week, month or whatever, they upload those leads to the corporate office. The leads can still be maintained on the remote workstation, but, the home office can evaluate them for forecasting and other analysis.

Approach

As with any other system, the software functionality should be the primary driver as to what system should be put together. The functionality of sales lead tracking software is different from many systems that may go into a data center. It is important that the projects aimed at decisions of what packages to implement, incorporate the sales force in their process. Because sales lead tracking is a relatively new concept, not many MIS people understand all the ins and outs. It is currently a specialty unto its own. Before implementing a new system, make sure that all the bases have been covered with an individual that knows these types of systems.

The functionality that needs to be present in sales force software is going to vary from organization to organization. There are some basic pieces that should, however, be present in every one of those systems. Most importantly, the system must be easy for the sales rep to use. This implies more than just user-friendliness. Flexibility, speed, and as few keystrokes as possible are all vital components of this. If the system is not easy to use, you will never get your sales reps to use it. The more the reps use the system the better sales results will be and the more the overall data will be helpful to management.

Some additional features/functions that should be present in a sales lead tracking package are:

- Ability to import mailing lists from third party or outside sources
- Creation of tickler/action fields to zoom in on only relevant data
- Ability to store free-form text easily along with lead information
- Ability to perform literature fulfillment and mailing labels for mass mailings
- Should be able to generate batch letters as well as customized letters on the fly
- Should have the ability to look at data a variety of different ways through keys
- Ad-hoc reporting features
- Flexible security so that there is no problem with reps stealing information from one another, but, management can view all relevant information
- Ease of interface into other existing data processing applications

These are just a few of the major things to look for in a sales automation package. There are many more things which could be of importance to your own shop, i.e., interface with graphics, audit trails, data dictionary, documentation...Just remember, flexibility and ease of use are of primary importance.

Conclusion

It is my belief that the time to automate your sales force is now. Those organizations that do will have a competitive edge over their rivals. Most of the other major functional areas in an organization have been automated, the sales force and lead tracking areas represent one of the last frontiers in data processing. The ability to get a handle on how your reps are performing as well as providing the sales force with good tools is a temptation MIS directors should take seriously.

Return on Investment Example

per $000 in Annual Sales Volume
(Manufacturer Profile Assumptions)

	Current Base Amount	(Choose one below) <----Increase in Sales Due to----> Automation 5%	10%	20%	30%	40%
Sales	$1,000	$1,050	$1,100	$1,200	$1,300	$1,400
Manf. Cost of Goods	500	525	550	600	650	700
Gross Profit (50%)	$ 500	$ 525	550	$ 600	$ 650	$ 700
Selling Expenses (20%)**	200	210	220	240	260	280
General & Admin. (Fixed)	200	200	200	200	200	200
Net Profit before Taxes	$ 100	$ 115	$ 130	$ 160	$ 190	$ 220
Net Increase over base	---	15	30	60	90	110
Percentage Increase over base profit due to Software	---	15%	30%	60%	90%	110%

** Worst case assuming all selling costs increase as the sales productivity increases. In actuality, some of these expenses will remain fixed at a lower level.

Return on Investment Example

Software Calculation Examples

(Manufacturer Profile Assumptions)

	<----Increase in Sales Due to Automation---->				
	5%	10%	20%	30%	40%
Case: $1,000,000 Annual Sales					
1. Increase in annual profit	15,000	30,000	60,000	90,000	110,000
2. Annual support fees	2,400	2,400	2,400	2,400	2,400
3. Software	14,400	14,400	14,400	14,400	14,000
(1. minus 2.) divided by 3. % = ROI	90%	191%	400%	608%	747%
3. divided by (1. minus 2.) times 12 = Payback in months	14	6	3	2	1

THE BIGGEST COMPUTER SECURITY THREAT

by Vladimir Volokh
VESOFT
1135 S. Beverly Dr.
Los Angeles, CA 90035 USA

The problem of computer security was definitely not invented by software vendors -- just read the newspapers every day.

Computer crimes come in different flavors:

* simply reading sensitive data (prices, customer lists, etc.)
* modification of data (payroll rates, shipping information)
* sabotage (viruses, time bombs, intentional system crashes)
* software theft
* unauthorized computer use
* defense-related crimes
* and more...

Security-minded authors have written many books, as well as articles in HP-related publications, on this subject; we at VESOFT became involved in the HP 3000 security industry in its very infancy, presenting computer security papers at HP conferences in Berlin (1981), Copenhagen (1982), Anaheim (1984), and, most recently, at the INTEREX security seminar in 1989.

And yet not every HP3000 computer is secure! The word **SECURITY** is misleadingly simple, simple enough to make many people think that they have adequate system security without fully thinking out what HP 3000 security really entails.

The issue of computer security is actually very complex -- it involves:

* physical security (guards, dogs, locks)
* system set-up (accounts, groups, users, capabilities, access, etc.)
* LOGON security
* file system security (why does MPE have a :RELEASE command?)
* IMAGE security (have you EVER changed your database password?)
* application security (who is allowed to print checks?)
* data encryption (fields, files)
* LOGOFF security (can people just walk up to an unattended terminal and use it?)
* batch access

* back-up and disaster recovery

* and more... much more...

System security is every bit as much a primary concern of any DP department as the actual applications running on the machine.

SYSTEM SET-UP

Look at your accounting structure first:

* What accounts do you have (check it by using :REPORT X.@)?

* What groups (use :REPORT @.@)?

* What users (:LISTUSER @.@)?

* Which capabilities do each of them have (SM, OP, PM)?

* What kind of access (Read, Write... -- for ANY, AC, GU...)?

* Are all of these entities passworded, or only some of them?

* Are some of the existing passwords too short or too obvious?

* How often are they changed (if at all)?

* How many various levels of UDCs are set on your system?

* And if you rely on them, how easy is it to bypass them?

LOGON SECURITY

Logon to the HP seems to be quite secure with ACCOUNT, GROUP and USER passwords. Or is it? Look carefully:

* MPE error messages at logon time are too *"friendly"*

* Passwords are **readable** combinations of up to 8 ASCII characters

* They are either easy to guess or difficult to remember -- and users write them down (sometimes even stick them to the terminal)

* They are often shared or simply disclosed

* Seldom changed

* If users use the session name (:HELLO MARY,MGR.PAYROLL), it only looks better -- the session name isn't enforceable and the password is assigned to user (MGR) anyway

* Yes, the MPE password is assigned, so account manager is the first suspect (and all SM users too)

* Is it easy to enforce shifts (time restrictions on logon)?

- Can payroll be run in the computer room (from LDEV 20)?
- Or can it be done on the weekend?
- Can end-users ever see ":"? What can they do then?
- What is better: to forbid most MPE commands via *"clever"* UDCs or to let users execute only some commands and subsystems?
- And if you have a logon UDC which brings them into an application, how about some other applications (e.g. HPMAIL), some utilities? Should users constantly change their logon ID?

REMOTE ACCESS

Remote access to the computer is common nowdays: dial-up, DS, NS...

- Who knows your dial-up telephone number? Your former employees, current employees, telephone company workers, HP SEs...
- Simple question: what to do if a person leaves the company? (Change all passwords on the system, unplug dial-up forever, request to change your dial-up number...)
- We have a horror story to tell you: one of our customers did change their dial-up number, but... the telephone company set call-forwarding onto it (you know the message -- *"The number has been changed, the new number is..."*)
- And if there are two or more computers linked together, can any programmer access the production HP/960 from the development HP/42?

LOGOFF SECURITY

Logoff is also a problem: you should realize that unattended sessions constitute a major threat to system security. The more sessions you have (it can be hundreds on XL) the less control you have.

- Remember that an unattended terminal is a convenient way for some people to use your system without logging-on.
- Also, if the session is left on after hours and keeps some files open these files might not be backed up.

BATCH SECURITY

Batch security is as important as on-line, but...

- MPE requires passwords to be included in job cards, so a typical job card looks like this :JOB FULLDUMP,MANAGER/SECRET.SYS/QWERTY
- This makes passwords easy to read by unauthorized people and difficult to change on a regular basis by people responsible for system security (it might be you)
- There are some other important things built into streams -- lockwords, database and/or application passwords, etc.
- The situation is somewhat better if all of your streams are in groups with X:ANY,R:GU

access -- but try to verify this

IMAGE SECURITY

IMAGE security had better be good -- that's where our most important data usually is. However

- Passwords (up to 63 of them) create the appearance of good protection of the base, sets, and entries
- But... these passwords are often built into sources -- intrinsic DBOPEN requires this; source code is compiled and guess what? IMAGE passwords are **never** changed! The situation is so bad and continues for so long, that HP users seldom recognize this kind of danger.
- It's even worse when using some application packages -- all customers of this package have the same password. Would you buy a car with the same key for everybody else who buys the same car?
- Some system managers sense something wrong in this area and set a lockword on QUERY. It's better than nothing, but what about other database retrieval tools or custom written programs?

FILE SYSTEM SECURITY

File system security in general is very important. A couple of questions come to mind:

- How many files on your HP are released?
- Even worse -- are these files in PM groups?
- And how do you :SECURE hundreds of them? Are you the *"creator"*?
- Which files were accessed on your HP over the weekend?
- How many programs, and which ones, have PM capability?
- Is it possible to :FCOPY the object code of your programs in ;CHAR mode and see all the 'built-ins'?
- Do you like the recent ACD (Access Control Definition) enhancement for MPE/V file system, which, in short, links particular users to the file?
- If so, before using ACDs, think about selection of these files later -- they will be as invisible as :RELEASEd files; think also about setting ACDs for groups of files, saving ACDs after editing text files, etc.

Having said all of the above let's ask ourselves:

What is **the biggest** computer security threat?

And it seems that the problem lies in the wrong approach to the risk management on the part of DP personnel. As long as system managers continue to count on users' ignorance, on end-users being *"good"*, on having only one dial-up line (yes, we've heard this one too) and such, company assets -- and some people's resumes -- will be in danger.

DEVELOPING CASE APPLICATIONS

Robert H. Ohlwiler

Quality Consultants Inc.
1775 The Exchange
Suite 380
Atlanta, GA 30339
(404) 980-1988

INTRODUCTION

This paper addresses the development of applications using Computer Aided Software Engineering (CASE) software. It defines the CASE environment and contrasts CASE development to traditional development methodologies. The importance of CASE to system maintenance is also discussed. Finally, recommendations for implementing CASE are presented.

DEFINING CASE

CASE Is A Philosophy

CASE is a corporate philosophy that imposes the engineering discipline on the development of application software. It embraces the entire range of the application life cycle from preliminary design through maintenance. Any tools that support the CASE philosophy can be considered within the CASE tool set. Integrated CASE Tools are tool sets that support the entire range of the application life cycle from design through development and continue to support the application modifications in the post installation or maintenance environment. The use of various CASE products is the means to managing the CASE philosophy. In this paper, CASE is addressing the complete system life cycle utilizing Integrated CASE Tools.

CASE Basics

I feel CASE would be more appropriately named "Computer Automated Systems Engineering" rather than "Computer Aided Software Engineering". I like the word "Automated" because CASE tool sets generally include code generators (screen generators, DBMS generators, etc.) that automate the coding function by generating programs (forms files, schemas, etc.) from design criteria. I like "Systems" over "Software" because people equate software with programs, and CASE is attacking the integrated application (or system) and consequently is automating the construction of applications more so than individual programs. These applications include screen generation, data base and file generation, and documentation generation, in addition to program generation.

CASE is often divided into "Upper CASE" and "Lower CASE". Upper CASE is generally associated with flow diagramming utilizing design methodologies such as Yourdon Diagrams, Warnier Orr Diagrams, Chapin Charts, Jackson Structural Hierarchies, etc. Lower CASE is generally associated with tools that eventually construct the working application. Lower CASE often includes integrating forms design, security, documentation, file construction, etc. into a controlled environment. The integration is accomplished by the use of a system inventory or repository.

The repository stores all the variables of the design and information about the system. A core component of the repository is the data dictionary. Ancillary

data dictionary information such as element help, data groupings, cross referencing, and data structures are also included in the repository. Prototyped screens, prototyped reports, processes, system structures, menus, security parameters, and any other system components that are collected by the CASE tools are stored in the system repository.

The repository is an important element of the CASE environment, it links together the preliminary design, the technical design, system construction and system maintenance. It is a powerful component of the development cycle because it provides the results of a stage of development (e.g. the preliminary design) to be stored in machine readable form for use in subsequent stages of development (e.g. maintenance).

IMPLEMENTING CASE

When Do You Need CASE

A CASE development approach makes sense when commonality is low and organizational impact is high. When commonality is high (e.g. Accounting Applications) you can not ignore the savings associated with package software. Too often management sees 4GLs and CASE products as a means to develop everything in house. The cost of developing your own application is significant compared to purchasing packaged software that has development costs shared by a large number of users. When organization impact is limited (i.e. it effects individuals or small work groups), you are better off to provide PCs and PC software tools like spreadsheets, word processors, graphics, etc. and let the individuals utilize those tools as they see fit. When commonality is low, organization impact is high, and economics justify the expense, you are forced to develop an application. If you can justify development, you can justify CASE.

CASE Requires A Significant Learning Curve

The fact that Integrated CASE Tools encompass the entire application life cycle, and that they automate the development process, implies that Integrated CASE Tools perform a lot of functions. Such a significant implementation of development tools is going to require a lot of learning and a substantial learning curve. On top of the significant learning curve associated with the CASE tools is the rethinking that must take place with disciplined development and joint application development (JAD). Consequently, the implementation of CASE needs to be carefully managed to assure an efficient and successful start-up.

CASE Requires A Rethinking Of The Development Process

CASE forces a team approach to software development. User involvement brings the technical and user team members into a closer and on-going contact. The system repository with its central source of data elements, file structures, stored prototypes and reusable processes forces the technical team members to coordinate their activities with the other team members. Programmers use

to coding independently without sharing or coordinating copylibs, subroutines, development tools, etc. are forced to be team members under the CASE philosophy.

With traditional development, corporate MIS standards are often guidelines that are optionally adopted on an individual by individual basis because enforcement is difficult to assure. CASE products standardize the structure of the design (e.g. Jackson Hierarchical, etc), the format of screens and reports, the method of controlling reusable processes, the data base schema, documentation format, on-line help functions, and virtually every standard initialized within the CASE tools. MIS departments that had no standards, or had standards that were adhered to occasionally, will have to retrain personnel to work in a more disciplined environment.

CASE tools include many productivity enhancement features such as preprogrammed building blocks, incorporated on-line help, etc. Just as programmers needed to think through how to best use the traditional tools (e.g. COBOL, Image, View, etc.) to provide users with the applications they needed, they must think through how to best utilize CASE tools to take advantage of their features as well.

CASE allows system developers to spend more time on design, and relieves them of the tedious coding exercises associated with system development. Consequently, debugging is often performed at the design level, and many things associated with coding, such as syntax errors, editor files, and JCL are no longer items of concern.

Developers are not tied to an application. The disciplined environment allows for programmer independence. Anyone familiar with the CASE tools will find it much easier to perform modifications on an application that is foreign to them without concern for a programmer's personal style of coding that is typical with third generation languages.

User Involvement In The CASE Environment

Traditional software development involves user management during the preliminary planning phase to "kick off" the development project. Many months later, a working system is presented to the user community, often to people not involved in the preliminary planning cycle. The results are often not on target, and a modification process begins to correct the system as best as can be expected so late in the development cycle, and with large project-to-date investments.

The term Joint Application Development (JAD) has been adopted to describe the user/developer team approach to application development. The CASE discipline facilitates user involvement throughout the application development. Users take on more development functions, and are continuously reviewing the evolution of the application.

A key ingredient of the CASE environment that supports user involvement is prototyping. Prototyping includes screen and report designs, menu and screen hierarchies, system structures, and partially working functions. Prototyping requires user involvement at the front end of the development cycle and continued involvement as the system evolves.

CASE IMPROVES THE QUALITY OF APPLICATIONS

Reduces Human Error

CASE code generators consistently generate error free code. With traditional source code level development, programmers spend considerable time correcting syntax errors and general coding mistakes. Often these errors will pass a syntax test, clean compile, and require extra testing time to locate and correct. Too often, many of these errors are discovered during the first six months after the system is running in production mode. These errors are very disruptive to the business. CASE reduces testing, user acceptance, and post implementation support times by reducing human errors in system development.

Staff Independence

In the traditional development environment, standards define what design documents, screen formats, reports formats, menu selections, on-line help, etc. should consist of and in what formats. However, these standards are generally difficult to enforce. Developing systems in the CASE environment guides and often forces the developer to work within certain standards. Also, machine generation of program code, screens, data bases and documentation eliminate the style and structure personalities of individual programmers. When these standards and machine generated components are in place, applications begin to look very similar. Programmers can easily initiate themselves to a new application, and enhance the application without their individual style and structure impacting future enhancements.

Smaller Project Teams

According to Brook's Law from his book titled The Mythical Manmonth[1], as people are added to a project team, the output per person declines. This is due to such things as communication requirements, project coordination, "stepping on each other's toes", etc. CASE promotes the inverse of Brook's Law by allowing smaller development teams due to increased individual productivity. As the number of people per project team is reduced, the productivity of the team members go up.

Improved Documentation

The quality of documentation is a difficult thing to manage in the traditional development environment. This is true because traditional development is document-driven. A preliminary design document is usually left in its finished

form while a detail design document is being produced that changes and adds components to the original plan. In turn, programming and user acceptance continue the evolution of the system without associated updates in either the preliminary or detail design documents. Finally, at the end of the project, when the developer's motivation is to wrap up and get to the next project, a reference document is hastily pulled together. It usually consists of best guess components with respect to what is needed to support the system.

Often the documentation step is de-emphasized by management because of management priorities (they are directed and measured on getting the system installed and bug-free), other project priorities, and emphasis on keeping within the budget (which is often exceeded at this stage of development). This situation gets worse as system maintenance and modifications take place. Consequently, the only accurate documentation is often source program listings and executable JCL.

CASE improves system documentation by "leaps and bounds". To begin with, CASE applications are prototype-driven rather than document-driven. The preliminary prototype is in machine readable format to accommodate expansion in the detail or technical design steps. The detail prototype is also in machine readable format and consequently is kept current as the system develops. This complete system updating continues into the modification and maintenance environment after the system is in production. One of the guidelines of efficient documentation is to document as you develop. This is based on the idea that the documentation you need during development is the documentation you need for maintenance. Prior to integrating documentation generators with development tools this great guideline was very difficult to implement.

System generated documentation has several other inherent advantages. First, the documentation is generated from the same repository, and consequently the same parameters as the generated application. This gives credibility to the accuracy of the documentation. Second, the documentation is easy to produce. Most documentation generators within the CASE product have a selection menu to produce the desired hard copies. Also, much of the documentation is available on-line directly from the repository. With the system set up to produce documentation as needed, the tediousness of an all encompassing document is reduced to a relatively simplified task. Third, the system documentation is consistent from one application to another, and you do not need to have indepth knowledge of the application to know where to locate system documentation.

Free Components With Every Application

I refer to free components as being features of the generated application that are built into the CASE development generator. A good example is on-line help to the element level that is a common feature in CASE products. Every generated screen has the ability to select a field on the screen that can jump to

a detail help information screen, and back to the point where the help was initiated. To add this capability to a COBOL level development would be very expensive and probably would not be included in the development. Other examples of what I call free components include such things as pop-up windows, security, print screen functions and graphic and spread sheet interfaces.

CASE SUPPORT OF THE MAINTENANCE ENVIRONMENT

According to studies produced by several defense companies[2], 65% of the total cost of applications over their entire life cycle are spent in a maintenance mode, while 35% of the cost is related to design, development, testing and implementation. Also, 80% of the system changes made during the development phase occur after approval of the original design. Consequently, about 93% of the cost of the system takes place in a maintenance environment. Couple these statistics with the fact that many system changes do not get approved because of complexity and cost, and it becomes clear that to reduce total system costs, CASE must address system maintenance.

Maintenance of systems developed in the traditional third generation environment have several inherent inefficiencies. First, third generation languages such as COBOL or Fortran facilitate individual personalities in organization, structure, and clarity of code. Second, maintenance is typically performed by programmers that were not on the original development team. Often they work around the original code when it is not clear to them and add their personalities to the code. To make matters worse, turnover is high due to the frustrating work environment. Patches are made on top of patches by multiple individuals until the systems are too expensive to support. In addition, as we discussed earlier, documentation is meaningless or non-existent. Consequently, systems developed and maintained by traditional methods become high-maintenance systems as they age.

Management of high-maintenance systems is difficult. Upper level management often does not relate to the fact that systems deteriorate over time. Maintenance programmers are hired based on their knowledge of the programming language, system software, hardware and general application experience. Their biggest learning curve is the application software, not so much because of what the software accomplishes, but because of unmaintainable code written by several previous programmers with varying styles and levels of experience. Not only is time not budgeted for learning the high-maintenance system, management does not prioritize maintainability enhancement projects high enough (relative to system modifications and new development) to correct the problem. As a result, the maintenance programmer is under pressure to make modifications to programs that are difficult to change and while doing so, eventually learns the code as best as can be expected from hands on experience.

The CASE development environment does more for the application mainte-

nance budget than it does for the development efficiencies. Improved maintainability starts with the fact that CASE tools support development standards. These standards include an application structure that is common from one application to the next. In addition, there is no patched code. Each time the system is regenerated, the code is regenerated from the design parameters. Personal style and structure is much more limited. Also, system generated documentation is consistent from one application to the next, and is accurate and up to date. As a result, the learning curve is limited to application features, which are typically easy to figure out. Once the maintenance person learns the CASE tools, they can maintain multiple applications effectively.

The CASE repository provides maintenance added value features, particularly with respect to reporting and global change capability. In the traditional maintenance environment locating all effected areas for a change (e.g. in a field size change for part numbers) is extremely difficult. The management of that change is enhanced significantly with a "where used" report. This report identifies every file, screen, report, and process effected by the change.

All of the features of CASE development are also available in the maintenance environment. Prototyping and increased user involvement reduce communication errors and increase user acceptance. All of the cost and time saving features of CASE development apply to on-going modifications.

MIGRATING TO THE CASE ENVIRONMENT

Migrating to CASE development is a significant undertaking for an organization. It requires management commitment and upfront investments in software, training, and inter-departmental team work. Based on our experience from implementing CASE at Quality Consultants, Inc., as well as several of our client's sites, we recommend the following guidelines, or rules to follow to assure a successful implementation.

- Rule number one is to get management commitment to implement a CASE strategy and assure appropriate resources are available for a successful implementation.
- Rule number two is to budget for training and not cut corners or underestimate the importance of training.
- Rule number three is to implement a small project first. The learning curve is best traversed in a simple environment which is completed beginning to end.
- Rule number four is to have a person that is experienced with the CASE tool set on board during the initial start-up. This person should be on board as soon as training is completed and remain on site for the first month or two. Your CASE tool set vendor should be able to assist you in finding a person that can fill this role.
- Rule number five is to identify a person to become the CASE product site expert. That person should be a fast learner and motivated to assume the position. They should be allocated more time and more indepth training with the experienced consultant.

REFERENCES

[1] Brooks, Fredrick P. Jr., The Mythical Manmonth, Addison-Weslen, 1974.

[2] "Programming Tools: Everyone's Getting into the Act," Business Week, May 9, 1988, Copyright 1988 by McGraw-Hill, Inc.

Managing performance issues in both MPE/V and MPE/XL

F. Alfredo Rego

Adager

Sun Valley, Idaho
83353-0030
U.S.A.

I have always dedicated myself (in a private kind of Technological and Managerial Olympics) to the constant quest for more performance. In the process, I have faced many challenging technical and managerial issues which are relevant to *any* application that runs on the HP3000 computer (whether Classic under MPE or Spectrum under MPE/XL).

Naturally, theories are nice *but* we need to implement and manage them to make them useful. I have been fortunate. Since I have enjoyed many different types of Hewlett-Packard computers as platforms for the various implementations of my theories, a bit of nostalgia is appropriate.

I learned my first lessons on Hewlett-Packard computer performance in 1974, when I worked on an HP2100 with RTE II at the Telephone Company in Guatemala. The challenge was to create an automatic billing system for international direct-dial calls. The resources were extremely limited by today's standards: 64k bytes of magnetic-core memory, 5 megabytes of disc (2.5 fixed and 2.5 removable), one paper-tape reader, one teletype with paper-tape punch, two 800 bpi tape drives (no read-after-write), 1 optical card reader, 1 CDC 600 lpm printer and 5 Datapoint 2600 terminals (with upper case only). HP did not make *everything* in those days!

From 1975 to 1986, I spanned the 70's and the 80's working under MPE on pre-Classic and Classic HP3000 computers. In the last few years, I have enjoyed the dramatic evolution of Hewlett-Packard's MPE/XL operating systems and HP Precision Architecture (Spectrum) hardware.

From a technical and managerial perspective, there are some fascinating questions: What have I learned in the last few decades? How does knowledge evolve? How can I apply this understanding in the decades ahead? In this essay, let's explore (in a nicely relaxed manner) *what* I have done, *why* and *how*.

Warm up

"Performance" is certainly a heavy subject. Let's just get the juices flowing and limber up stiff intellectual joints, as we would do before embarking on any strenuous physical activity. Let's also stretch our mental muscles by mentioning other important concepts which we cannot afford to ignore while we strive for performance, such as: clarity, relevance, ease of use, reliability, functionality, robustness, accuracy, maintainability, precision, privacy, security, and so on.

Let's take reliability and robustness as examples. A reliable and robust system requires a significant amount of software effort to check for (and hopefully correct) all kinds of unacceptable conditions. Some people might think that such effort detracts from the system's performance. I beg to differ: Designed-in reliability improves overall performance, because the system does not have to pick itself up all the time as it "walks" (or, rather, "stumbles") towards its objectives. Lots of *redundancy* is the price we pay for reliability.

Let's consider the computer's relationship with the human user as another illustration. "Performance" has but one final motivation: to improve the productivity of the end user. A machine that performs miracles but requires gurus does not perform as well as a less lofty machine that allocates a lot of its power to communicate nicely with the end user.

A powerful interface allows the user to specify, precisely, what the computer is required to do. Ideally, the system should immediately catch any ambiguous or potentially damaging requests from the user *and* enter into a friendly discussion before a serious misunderstanding develops. This way, the machine is less prone to go off in the wrong direction to do something inappropriate blazingly fast.

An intelligent user interface requires a lot of software and computer juice. It takes a lot of MIPS for the computer to reach the conclusion that a message like "Hey, you!" is in order. Particularly if the user gets the message on the spot, in time for corrective action.

On the other side of the interface coin, the computer should communicate to the user, in clear terms, the results of its tasks. Instead of a 6-inch thick report of *all* widgets, the user will be better off with a 1-page report of those widgets that, given some quick action by management, will increase profits by 63%. This way, the user is able to make sense of the whole thing in less time, with less frustration and with better understanding (so the user does not go off in the wrong direction to do something inappropriate).

Perhaps we must accept different standards of "computer performance" when we are dealing with a human and when we are dealing with a machine.

I believe that raw performance is a worthy goal when dealing with raw bits and bytes, with operating-system intrinsics, and with any machine-oriented aspect of computing. But let's keep our perspective and let's not get lost in the game of "performance per se". We can spend our life saying "let's see how many times we can run this loop in a second" instead of thinking "let's see if we can eliminate this loop while still performing the same function".

Running

Let's remember a strange computing paradox. Running *any* program on *any* computer (even the world's most efficient program on the world's fastest computer) has a measurable negative effect: the net result is a degradation of the computer's performance! At first, we don't notice any difference. But as we slowly add more and more running programs, the machine

eventually and suddenly comes to a dramatic and humiliating standstill. This is the infamous "knee in the curve" that we see in slides during HP performance presentations.

I became painfully aware of this back in 1975, when running on a venerable Pre-Classic HP3000 CX computer (which we brought from Guatemala to Sun Valley and scheduled for an appearance during the INTEREX 1989 San Francisco Conference). Since then, I established a policy of having at most one programmer per computer (in my personal case, I feel quite satisfied about the fact that I have three computers at home, two Classics and one Spectrum).

The same holds true for you: get as many computers as you can afford, to spread the load!

Inertia

Newton said, "*inertia* is a property of matter by which it remains at rest or in uniform motion ... unless acted upon by some external force". Now, after having warmed up, we can appreciate the significance of Newton's insight regarding our own bodies, celestial or not: it takes a while and some effort to get going.

The same happens to computer entities. Every time you use ":run", or "dbOpen", or "fOpen", or "hpfOpen", or ":prep", you create a minor crisis in your computer system, as it goes through the anti-inertial motions of *causing* something to happen. By the same token, you cause great grief whenever you use ":abort", or "deAllocate", or "dbClose", or "fClose", as the computer wraps up a few things on your behalf.

The moral? Don't spend energy working against inertia. Don't call on any resources until you absolutely need them. And once you have the resources, don't allow them to remain idle: exploit them at the maximum and then let them free so somebody else may enjoy them.

I have a personal example to illustrate the power of inertia. In 1985 and 1986, I met with some top-level managers at Hewlett-Packard regarding an enhancement request for IMAGE that would cure a chronic case of *inertiae databasis*. I also expressed the request at Doug Spreng's session during the Washington 1985 INTEREX Conference. In those days, I was told "not to worry", since that enhancement was part of the new HPIMAGE. Now that TurboIMAGE/XL is "in" and HPIMAGE (for the HP3000) is "out", we seem to be back at square one.

The inertial problem: Currently, if you want to change the value of a critical (search or sort) field in a detail entry, you have to do two high-overhead operations: (1) DBDELETE the entry and (2) DBPUT the entry again after having changed the values in your program buffers.

This approach ("the way we have always done it") causes some horrible happenings.

(1) As you delete the entry with DBDELETE: The de-linking of all the chains where the entry is a link, the updating of the FreeEntry count (which is incremented by 1) and the addition of a new link to the FreeEntry list.

(2) As you invoke DBPUT to re-add the recently-deleted entry: The re-linking of all the entry's chains, the updating of the FreeEntry count (which is now decremented by 1 to go back to what it was in the beginning anyway) and the deletion of the newly-added FreeEntry list link!

This is not very elegant. Not to mention its negative impact on performance. A solution: Enhance DBUPDATE (with a new mode, perhaps "2", to preserve the current behavior of those millions of current callers of DBUPDATE with mode 1). DBUPDATE mode 2 would allow

changes to critical detail fields. At worst, if you changed all the critical fields for all the paths, DBUPDATE would have to de-link and re-link all the entry's chains (but it would not have to update and re-update the FreeEntry count and it would not have to link and de-link the FreeEntry list). At best, DBUPDATE would have to de-link and re-link only one chain out of 16, leaving the other 15 untouched, for a well-connected dataset.

Several people have asked me, since 1985, why I bother banging my head against HP's inertial wall instead of writing my own replacement for DBUPDATE, which I could then ship to my customers. The reason is simple: This is an enhancement to IMAGE that *only* Hewlett-Packard can do. Anybody can modify the *static* database structures (such as root files and datasets) but only Hewlett-Packard can modify the *dynamic run-time* database procedures (such as DBUPDATE) and structures (such as DBCB's, DBG's, DBUX's, or whatever). So, once again, I bring this topic up, hoping that some perceptive high-level Hewlett-Packard manager will pick it up!

Workout

Now that we have warmed up with a few casual topics, we are going to increase our heart rate by touching on a few controversial subjects. Let's begin with trying to define "performance".

What do we mean by "performance"?

There are, at least, two basic issues to consider: Throughput and response time. We, obviously, want to maximize throughput and minimize response time. Unfortunately, when we have *many* concurrent users, we usually end up optimizing a few at the expense of the others. If you ask "the few", they will tell you that the computer's performance is terrific. If you ask "the others", they will tell you unprintable things.

There are so many ways to define and measure performance! In your particular case, performance should be measured according to *your* definition, which ultimately depends on the productivity of *your* end users, following priorities specified by your management. One person's performance junk is another person's performance treasure. That's why this is such a touchy subject!

Consider the following: (1) a race car, with lots of horse power and raw speed which needs a smooth road surface; (2) a small 4-wheel-drive vehicle that can climb volcanos; (3) a mountain bike with just human power and very low gears and knobby tires; (4) a small, fragile helicopter that can land on delicate cornices at the top of snowy peaks. Each of these vehicles is a specialist, infinitely better than the others at a given task. Which "performs" better? All of them do, and none of them do! It all depends. The same applies to computers!

Performance and good citizenship

We have all been passed on the highway by a performance-oriented driver who almost blew us off the road. I don't believe that performance is incompatible with good manners toward fellow users of the same resources (be they roads, discs, memory, or whatever). The fact that

Spectrum HP3000 machines have much more memory (and neat facilities such as mapped files) does not mean that we should simply grab all the resources. We should still be considerate and use only the resources that are necessary (and sufficient) to achieve decent performance for ourselves and for our fellow users. From this perspective, we should try, even within the riches of an MPE/XL system, to follow the discipline that MPE V *imposed* on us.

Conceptual models: Bit maps and stacks

A good conceptual model is a map that will guide you through the performance wilderness. A lousy conceptual model will get you hopelessly lost in time-dated performance trivia.

For instance, instead of moving massive amounts of data around, you may choose to appoint *ambassadors* or *class representatives*. The ultimate example is a bit map. On a Classic (MPE) HP3000, you can place more than half a million bits in a single extra data segment. With a smart model, you can use the bit map to find out what you need about an entity in, at most, one disc access (to virtual memory, if your extra data segment was swapped out). Even better, since you will probably want to know a lot about lots of entities, the memory manager will automatically keep your extra data segment resident in memory and you will be able to get answers relating to hundreds of thousands of entities without a single disc access! Compare this with a model that would require you to wade through 2000-byte entries in a file instead of breezing through tiny bits in a bit map.

The idea is that it's a lot easier to shuffle symbols than to shuffle the entities represented by the symbols. The trick is to come up with smart symbols and good paradigms. Fortunately, Hewlett-Packard has championed some excellent ones, such as stacks and RPN.

Stack architecture and Reverse Polish Notation (so beautifully implemented in HP hand-held calculators and in the Classic HP3000 under SPL) are very powerful tools that you should consider keeping in your bag of tricks. The fact that the Spectrum HP3000 computers are register-oriented does not mean that you cannot implement your *own* stacks. As a matter of fact, I have been successfully using, since 1987, software-implemented stacks that are amazingly effective on *both* Classic and Spectrum HP3000 computers. I would go insane if I even tried to model the complex things required when transforming databases with recursive functions (such as adding a path to a non-existing dataset that will require fields whose items are also non-existing). I wrote the stack-management modules once and then forgot about them. Now I can spend my time and energy thinking about more useful things, such as mapped files.

Mapped files and extra data segments

If your application runs exclusively on the Classic HP3000, by all means use extra data segments for buffers. If your application *also* runs on the Spectrum HP3000, you may choose to take advantage of mapped files for some cases and you still may decide to continue using extra data segments for other cases. As a matter of fact, MPE/XL simply offers you more choices in buffering technologies. Depending on *your* quickness of mind, you can make one non-obvious technology (for instance extra data segments) perform much faster than some other obvious technology (for instance mapped files). Don't be swayed by buzzwords: after all, you measure performance with numbers, not with words!

Parallel processing

Serial processing is the simplest kind of scheduling, but it may also produce the slowest results from a performance viewpoint. Parallel processing may become a scheduling nightmare, but it may be the only way to gain significant performance improvements.

Scheduling involves the allocation of resources such as time, memory space, disc space, central processors, peripheral processors, etc. Parallel processing involves the *concurrent* allocation of resources. Not all parallel processing is necessarily good. Witness a street intersection: if we "allocate" all vehicles from all directions into the intersection without any rules at all, we will certainly get a messy collision. This analogy has been present in the minds of most implementors of parallel technologies, and the common vocabulary reflects it (for instance, the term *semaphore* means "traffic light"). As if scheduling vehicles through a busy intersection were not challenging enough, we must still allow for *priority* vehicles (such as ambulances or police cars) that, theoretically at least, should breeze through intersections, even against the flow controlled by automatic traffic lights.

Think of an orchestra conductor who has quite a few instruments as *resources* (assuming that they are all played by competent musicians, which is another story in itself). Most of the time, the composer calls into *active duty* only a small subset of all possible instruments. And, most of the time, not all instruments in this small subset are playing at the same volume with the same intensity. You might say that instruments have various priorities, which range from *nothing at all* (even less than *pianissimo*) to *all out* (*fortissimo*). The object of the game is to obtain the best performance possible. The person in charge of ensuring the necessary *dynamic* allocation of resources is the conductor, who should certainly follow the original specifications of the composer.

Absolutely the same holds true for computers! You can design systems that take advantage of *many* concurrent processors focused on various aspects of the same complex task, or you can design systems that present a single processor with many tasks to be performed on a given MultiBuffer that you have brought from disc to memory. In any case, you want to avoid extra "notes" and extra "musicians". If you can accomplish a task with one disc access, please do not use 20. Even better: If you can accomplish a task without going to disc at all, please don't go to disc at all!

Parallel processing implies organization of tasks before executing them. This brings us to the topic of normalization, which I have covered in previous papers. Let's review.

Normalizing

I have established this simple definition of *normalizing:* "Put together those things that belong together and separate those things that don't belong together".

The methods for achieving the various normal forms in database theory apply this simple definition in a step-by-step fashion.

Normalizing applies to time as well as to space. Note that *separating* can sometimes be easier than *consolidating* (two incompatible things can be in the same place at different times or at the same time in different places).

"Sorting things out" usually refers to the process of normalizing. Getting organized before embarking on a task usually minimizes wasteful operations. I always try to consolidate as many operations as possible while scanning a disc file. (See "Parallel Processing" above).

The normalization of time involves deciding the sequence of operations that has the highest likelihood of minimizing the total elapsed time. I think it's impossible (and undesirable) to eliminate all backtracking, but constant backtracking is a sign of poor time normalization.

I became fully aware of time normalization after I read an issue of INTERACT in early 1983. Before then, my software was fairly primitive in this regard. Thereafter, I spent a lot of effort to improve its interface (by allowing automatic recursive operations when a user request finds a missing prerequisite entity) and its efficiency (by designing a dynamic scheduler that minimizes backtracking during the actual mapping of database files). Implementing this all-important aspect of computer software is no easy task. It took me five years to design and construct my version!

Caching, monitoring and self-tuning of intensive I/O operations

By definition, an intensive I/O (input/output) operation spends a lot of time thrashing around between main memory and disc (not to mention other, even slower, peripherals). Instead of covering significant ground, the application just seems to run on a treadmill.

One obvious solution for this performance bottleneck is to eliminate (or at least to minimize) the unnecessary transfer of data, back and forth, between memory and disc. This is the idea behind the concept of caching.

There are all kinds of caching schemes: central processor caches, MPE caching, disc-controller caching, and so on. I certainly like all these caching technologies and I use them whenever it is appropriate. But there is one kind of caching that has turned out to be the most effective, according to our measurements in the Adager Labs: Application-defined disc caching. Central processors, operating systems and disc drives do not know the application; therefore, they can at best guess what is going to happen next. The application itself, on the contrary, knows *exactly* what is going to happen next and can prepare for it by flushing cache buffers or by increasing/decreasing cache buffers or by performing any of the various caching operations.

An often-overlooked class of intensive I/O operation is *remote* file or database access. In this case, the thrashing happens between (or among) processors that are connected through some sort of communications link. You can bring two connected Series 955's to their knees (and you can allow two Series 37's to fly with the *same* task) if you do things stupidly on the 955's (and smartly on the 37's). It's as simple as that!

As an example of bringing two machines of *any* kind to their knees, take this task: "Access a remote database and report those customers who have not bought anything in the last 90 days". A not-so-smart way to perform this task is to run a local program that will examine *all* the records in the remote database (while passing them to your local computer through the communications line). If the entries of interest to you are relatively few, you will have wasted a lot of your preciously-narrow bandwidth transporting junk. A smarter way would be to run a *remote* program that will still have to access all the remote records (remotely for you but locally for the remote program). This is actually not too bad, since the communications line will not be touched at all while the remote program goes through entries that do not interest you. If and when it finds qualified entries, the remote program will send them to your local computer.

In this example, the performance figures are vastly different, depending on your choices of what to do locally versus remotely. It is amazing that you have basically the *same resources*. You are just orchestrating them differently!

Batch processing and sorting

Accessing a file *randomly* is an incredibly expensive operation, if we have to access *most* of its entries to accomplish a task (which is typical of *batch* processing).

Sorting is a way to reduce a file's randomness according to *one* dimension (a primary sort key and optional secondary keys). As a side effect, we usually increase the file's randomness according to its other dimensions, if any (the other non-key fields).

The HP3000 computer has been blessed with a good sort subsystem. Unfortunately, in the Classic (MPE) environment, programs with large data stacks did not have enough space to take advantage of direct calls to the sort intrinsics. Fortunately, several people developed ingenious ways to work around these limitations. In the Adager Labs, we developed an internal subsystem called "fwSort" (in Fred White's honor, since he worked out all the algorithms). Other companies have developed commercial sort subsystems that they market to the general public. And, naturally, Hewlett-Packard is constantly working to improve the performance of the HP3000 System Sort. So, there is no excuse for not sorting in batch mode!

A small ancient computer (such as a Series 44) may outperform a big modern computer (such as a Series 925) in batch mode *if* a big file is nicely sorted on the Series 44 and hopelessly randomized on the Series 925. You would be using the 44's miserable MIPS in an effective way and you would be wasting the 925's awesome MIPS thrashing all over the place. Remember the tortoise and the hare? Complacency is a fatal flaw!

OnLine Applications and their data structures

Accessing *all* of the entries of a file sequentially is crazy, if we only have to access *a few* entries (which is typical of *OnLine* processing).

My definition of *OnLine* is: "If you can answer a question over the counter or over the telephone without first having to say *please have a seat* or *I'll call you back*, then you have a true OnLine application".

For OnLine applications, sorting large files on the fly is out of the question, at least with the current sorting technologies, even though they can sort millions of entries in a few minutes. You would still have to say "please have a seat" or "I'll call you back", and you would not have an OnLine application any more. For OnLine applications, we must use techniques based on hashing, trees, lists, stacks, and mixtures of these and other data structures.

A humble computer (such as a Series III) may outperform a fancy computer (such as a Series 955) in OnLine applications *if* a big file is nicely structured on the Series III and carelessly disorganized on the Series 955. You would be using the III's lowly MIPS in an effective way and you would be squandering the 955's magnificent MIPS by sequentially scanning the whole file whenever you need to access a single entry. Remember David and Goliath? Self-satisfaction accompanied by unawareness of actual dangers or deficiencies is catastrophic!

Performing a square dance on a round table?

We can now see that "performance" is a relative thing. Using sophisticated data structures intended for OnLine use (such as IMAGE databases) in an exclusively batch environment will degrade performance and will waste valuable disc space. By the same token, using sophisticated sorting techniques intended for batch use in an exclusively OnLine environment will drive your waiting customers towards your competition.

Remember the distinction between "efficiency" and "effectiveness". You should be effective (i.e., you should do the right thing) and you should be efficient (i.e., you should do it quickly). You must not fall into the trap of being efficient while doing the wrong thing (or being inefficient while doing the right thing).

Cool down

It is now time to decrease our heart rate by mentioning a few calming items. After all, we could go on forever and we have to stop at some point, before we collapse from exhaustion.

A cool issue nowadays is migration from Classic to Spectrum machines. My experiences of 1987, 1988 and 1989 can be summed up by a few questions and a few answers to these questions.

Where do we stop? How much is enough? Compatibility mode? Object code translation? Native mode? Optimizing compilers? Before you worry sick about getting into these kinds of things, I would recommend that you check for the less esoteric kinds of things that we mentioned in this essay. Do a lot of macro optimization of performance *before* you fiddle with micro optimization. Take an infinite loop as an example. You can certainly execute an infinite loop most efficiently in native mode, but doing so would be even more stupid than executing it (less efficiently) in compatibility mode, because of the higher expectations.

An amazingly simple progression that has worked wonders for me is:

(1) Move your software from the Classic HP3000 to the Spectrum HP3000 in compatibility mode. Hewlett-Packard did a *fantastic* job with compatibility mode.

(2) Use the Object Code Translator to increase the performance (and the size!) of your programs. Hewlett-Packard did an unbelievable job with OCT.

(3) Incorporate specific procedures in native mode to be invoked *automatically* by your software when it detects that it is running on Spectrum hardware. Don't bother to do this for rarely-used stuff: Do your native-mode song and dance for those operations that you will perform a million times on raw bits and bytes.

(4) Go fully native with your current programs if you have the spare programming resources!

(5) Write your new applications in native mode, but don't forget to use sound management practices. The fact that you are taking advantage of Hewlett-Packard's new Precision Architecture hardware, new Operating System software and new optimizing compilers is not a license to kill. **Don't let your performance guard down, even when you have MegaMIPS at your disposal!**

ABSTRACT

TUTORIAL - PROJECT MANAGEMENT

Lewis R. Dalrymple
The City of Tempe
120 E. 5th Street
Tempe, Arizona 85281

Project Management in the HP3000 Environment is a tutorial covering a wide range of project management topics. This tutorial specifically addresses both hypothetical and real projects, using the 'Workbook Concept' for planning, implementation, and control.

The presentation is divided into three (3) parts, allowing participants to interact and develop a project:

PART 1 - The Basics of Project Management
* What is a project?
* What is Project Management?
* Project Management History.
* The Workbook Concept in Project Management.

PART 2 - Tools for Project Management
* Specification Sheets
* Task Planning/Assignment Sheets
* Network Development Diagrams
* Milestone Schedules (Gantt Charts)
* Resource Planning Sheets
* Project Documentation/Control Forms

PART 3 - Developing a Project
* Writing the Specification
* Planning the Project
* Creating the Task List
* Developing the Network Diagram
* Selecting the Project Team
* Assigning the Tasks
* Creating the Schedule
* Determining Resource Requirements
* Project Implementation and Control

Each participant is provided with a comprehensive workbook filled with many practical suggestions and examples of methods and documentation for managing any project large or small.

PROJECT MANAGEMENT

Overview

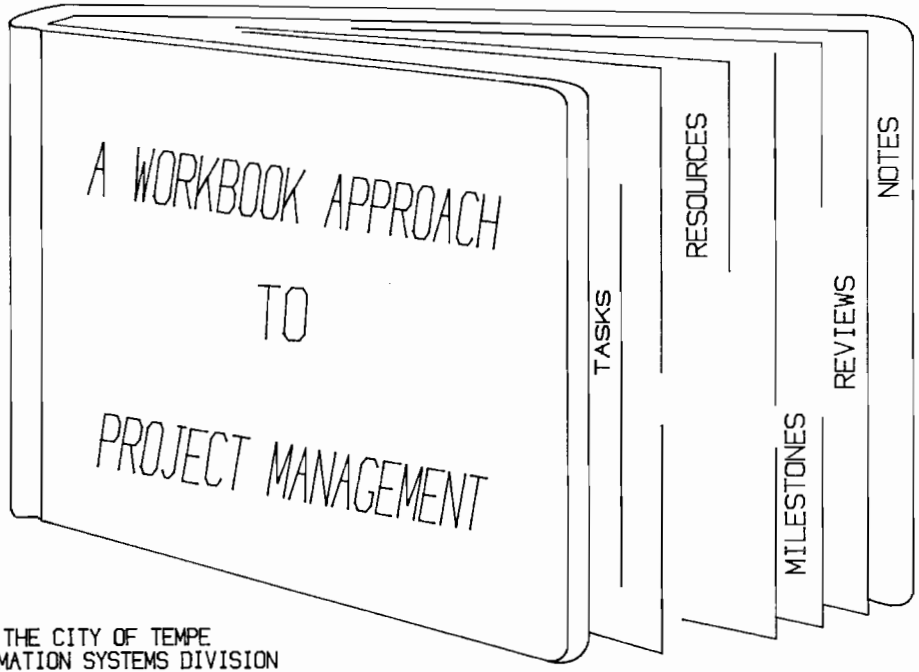
Mangaging a project means different things to different people depending on their background, training, and most importantly their general attitude.

Dramatic improvements in hardware and software capabilities of computers in the past few years have steadily increased the complexity of system development projects. These projects are no longer confined to any specific department, but have become dependent upon several departments and levels of personnel.

Developing project teams and managing such projects has become a challenge to all levels of management.

We are proposing only one of many project management approaches to meet this challenge.

PROJECT MANAGEMENT - TUTORIAL



THE CITY OF TEMPE
INFORMATION SYSTEMS DIVISION
LEWIS R. DALRYMPLE
1989

STUDENT WORKBOOK

TABLE OF CONTENTS

AGENDA

PART - 1 "The Basics of Project Management"

Presentation:	35 Minutes
Questions and Answers:	10 Minutes
Break:	10 Minutes

PART - 2 "Tools For Project Management"

Presentation:	35 Minutes
Questions and Answers:	10 Minutes
Break:	10 Minutes

PART - 3 "Developing a Project"

Presentation:	40 Minutes
Questions and Answers:	15 Minutes

PART 1

BASICS OF PROJECT MANAGEMENT

BASICS OF PROJECT MANAGEMENT

* WHAT IS A PROJECT??
* WHAT IS PROJECT MANAGEMENT??
* PROJECT MANAGEMENT HISTORY
* THE WORKBOOK CONCEPT

What is a Project??????

A Project is a well defined effort undertaken by a formally established team that is expected to produce specific results in a given environment at predetermined times.

The key characteristics of a Project are:

* It has a beginning and an end.
* It uses multiple, finite resources.
* It has an objective.
* Its progress can be measured.
* It requires a Project Leader and a staff.
* It must be planned.
* Its performance must be reviewed against the plan.
* It is affected by internal and external forces that must be identified and addressed.
* It is a group of sub-projects.
* It is not unique.

KEYS TO A PROJECT

IT HAS A BEGINNING AND AN END.

IT USES MULTIPLE, FINITE RESOURCES.

IT HAS AN OBJECTIVE.

IT REQUIRES A PROJECT LEADER.

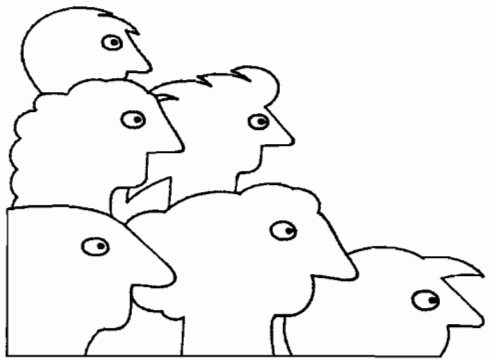
IT MUST BE PLANNED.

ITS PERFORMANCE MUST BE REVIEWED AGAINST A PLAN.

IT IS AFFECTED BY FORCES THAT MUST BE IDENTIFIED AND ADDRESSED.

IT IS A GROUP OF SUB-PROJECTS.

WHAT IS PROJECT MANAGEMENT???



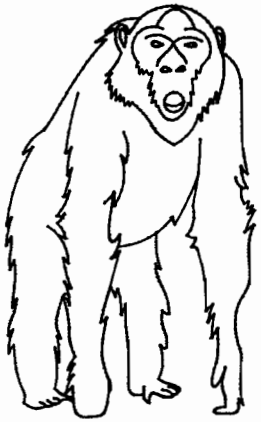
Project Management ???????

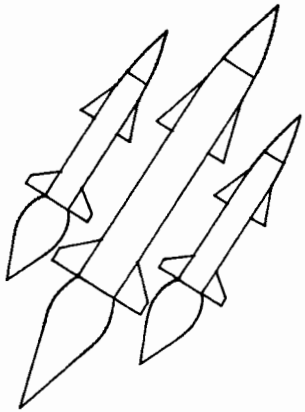
Project Management is the planning, directing, and controlling of resources to achieve established goals.

Although originally created to manage large and complex projects, it is now successfully used in all types and sizes of projects.

Project Management means temporary teams of people devoting skills to accomplish project objectives. These teams are led by the Project Manager, who ensures the successful completion of the project and supervises the team leader members.

Project Management is most effective in organizations or departments that perform temporary activities (ie. engineering, research and development, data processing, construction, etc.).





PROJECT MANAGEMENT HISTORY

PROJECT MANAGEMENT HISTORY

I will begin by giving you a little background on just how and when Project Management really started.

The Project Management-Concept was born back in the middle 1950's. It was the idea of 'running a business within a business'. That is to say, you could use personnel already assigned to a specific job (ie. engineering, purchasing etc.) to become subordinate players in a 'background job (project)' and also be responsible to another person (ie. the Project Leader) at the same time.

Well, this went against all of the traditional 'schools' of management, and received significant opposition from many areas.

P. O. Gaddis was one of the first to describe to the business world the functions and qualifications of Project Management in his article - The Age of Massive Engineering - published in the Harvard Business Review, during January and February of 1961.

PROGRAM PHASES

1) Definition of Problem

2) Analytical Solution

3) Mechanization

4) Verification

By this time, Project Management was an adolescent, not very sophisticated, but could read and write, however, the elders were very worried about its future.

Rapid technological advances in submarine/missile and space programs required 'new' techniques to implement and control complex projects.

This new hierarchial structure was not only acceptable to the project managers, but also seemed self-evident from the inherent complex nature of the systems/projects themselves.

It was soon found that every program had four phases:

1) Definition Of Problem
2) Analytical Solution
3) Mechanization
4) Verification

Such requirements indicated the need for many individuals/ groups which cut across firmly entrenched and traditional departmental boundaries. This created many problems for the new 'project manager', however, the end results would speak loudly and firmly for the new methodology.

In 1958, PERT (Program Evaluation & Review Technique) was introduced into the Polaris submarine/missile program during the late MECHANIZATION, early VERIFICATION phases.

It was aggressively implemented by the Navy, and within a year was in successful use.

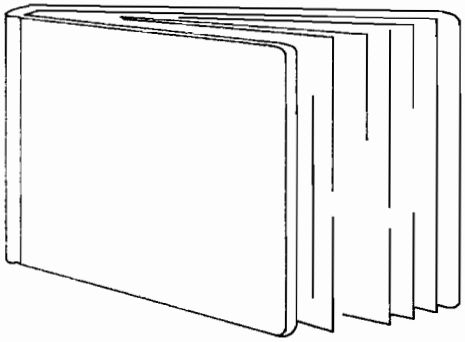
In 1960 PERT was tried in another Navy weapons project, in the DEFINITION OF PROBLEM phase, and was an immediate success.

Soon thereafter, the Navy (PERT) Project Management System was adopted by the Department of Defense, NASA, and many other defense contractors.

Project Management had now 'come of age'.....

In 1964 PERT/COST became the standard for DOD and NASA.

The PERT concept continues, in one form or another, in nearly every project designed and implemented today.



3017 - 18

THE WORKBOOK CONCEPT

WORKBOOK CONCEPTS

* STANDARDIZED WORKSHEETS
* DOCUMENTATION EXAMPLES
* SYSTEMATIC TECHNIQUES
* PROJECT FLOW CONTROL

The Workbook Approach

The Workbook Approach to Systems Analysis and Project Management is not new and certainly is not my creation.

I first became aware of this approach in the late 1970s when I used the Systems Analysis Workbook Guide, developed by Robert D. Carlsen and James A. Lewis, in developing software systems.

I further refined their idea, and have been using it ever since.

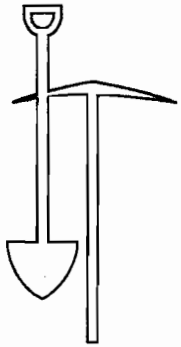
The book provides a uniform, sys
tematic technique for assisting in the performance of all steps of a project through the use of a standardized workbook.

The workbook contains a set of worksheets and documentation examples flexible enough to meet the special requirements of most projects, large or small.

The workbook is only one of many acceptable methods used in managing a project. The method you use on any given project should be the best suited to your needs and the overall requirements of that project.

Regardless of the method used, the workbook will provide a unique insight into Project Management and will serve as guide in planning any project.

PART 2



TOOLS FOR PROJECT MANAGEMENT

TOOLS FOR PROJECT MANAGEMENT

* SPECIFICATION SHEETS
* TASK PLANNING/ASSIGNMENT SHEETS
* NETWORK DEVELOPMENT DIAGRAMS
* MILESTONE SCHEDULES (GANTT CHARTS)
* RESOURCE PLANNING SHEETS
* PROJECT DOCUMENTATION/CONTROL FORMS

The "Tools", as we describe them, are the worksheets which ultimately make up the "Project Workbook".

We have included many different worksheets to be used in managing a project, and will describe how each is used in planning, implementing, and controlling your project.

SYSTEM/PROJECT SPECIFICATIONS

Two Project Specification worksheets and a System Specification worksheet are the first fact-gathering aids in the Workbook concept.

These sheets are filled out in the first stages of your fact gathering mission. System Specification worksheet, Figure - 1, provides general areas for required information.

Objectives required in the specification can be selected from the Objectives Considerations Checklist, Figure - 2, which provides a general list of possible items to be considered.

The heading data on the first Project Specification worksheet, Figure - 3, describes the general area to be studied, and provides space for recording the name, etc. of the senior individual (chief contact) for the project.

The main body of this form provides space for listing specific items to covered.

The second Project Specification sheet, Figure - 4, provides space for documenting other important items, such as; schedule considerstions, location, etc.

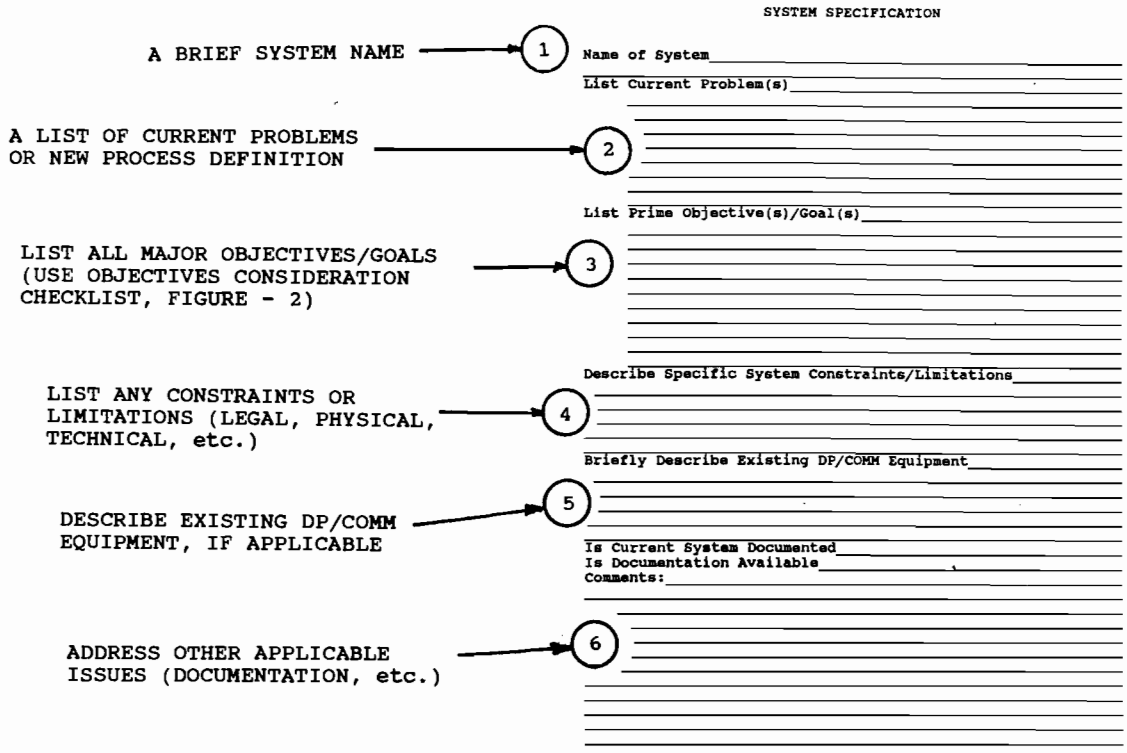

SYSTEM SPECIFICATION
A BRIEF SYSTEM NAME
1
Name of System
List Current Problem(s)
A LIST OF CURRENT PROBLEMS
OR NEW PROCESS DEFINITION
2
List Prime Objective(s)/Goal(s)
LIST ALL MAJOR OBJECTIVES/GOALS
(USE OBJECTIVES CONSIDERATION
CHECKLIST, FIGURE - 2)
3
Describe Specific System Constraints/Limitations
LIST ANY CONSTRAINTS OR
LIMITATIONS (LEGAL, PHYSICAL,
TECHNICAL, etc.)
4
Briefly Describe Existing DP/COMM Equipment
DESCRIBE EXISTING DP/COMM
EQUIPMENT, IF APPLICABLE
5
Is Current System Documented
Is Documentation Available
Comments:
ADDRESS OTHER APPLICABLE
ISSUES (DOCUMENTATION, etc.)
6

FIGURE -1

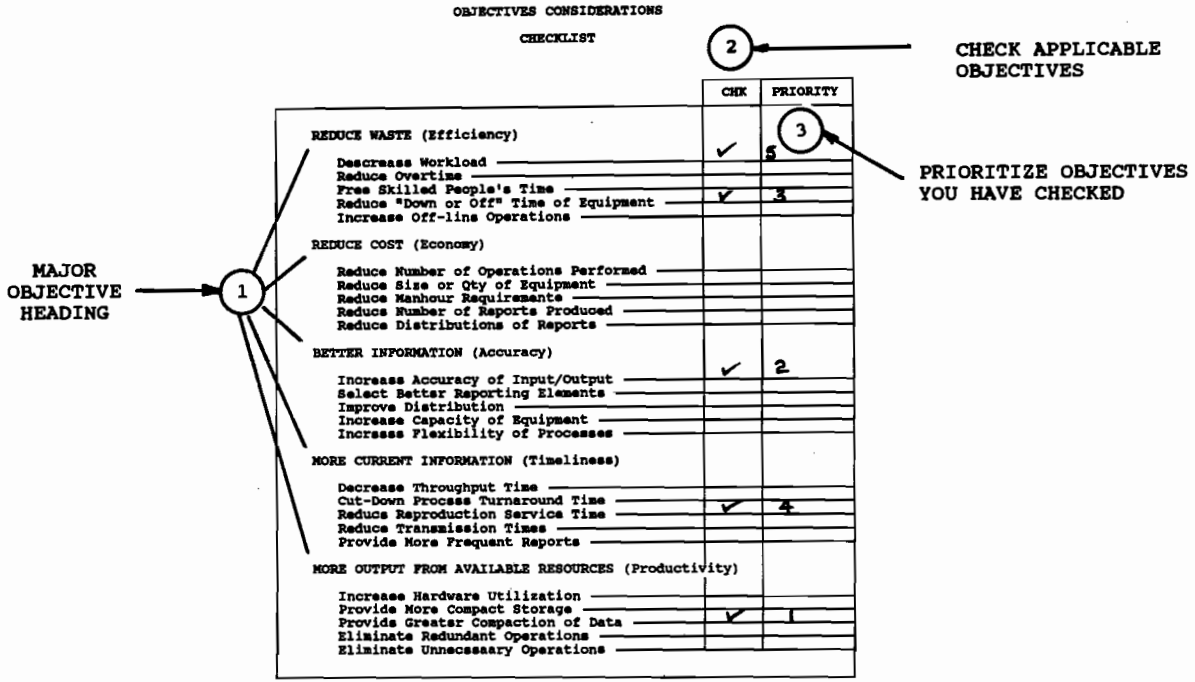


FIGURE - 2

THIS INFORMATION MAY NOW BE USED
TO COMPLETE THE SYSTEM SPECIFICATION (FIGURE -1)

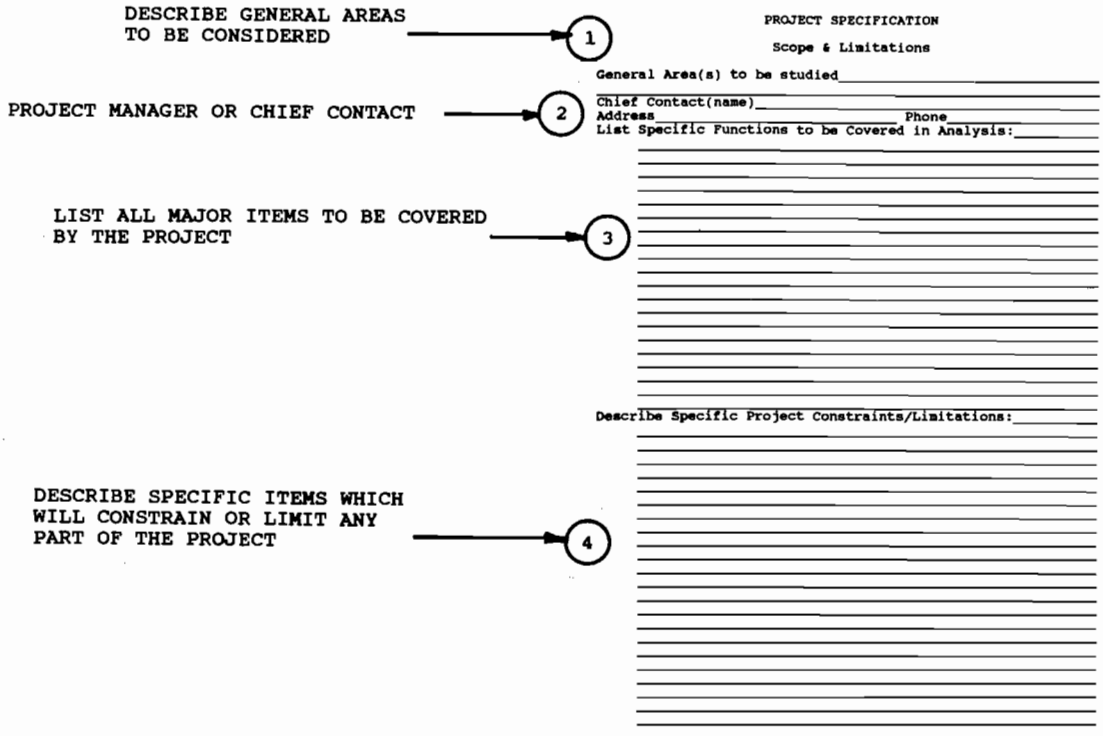

DESCRIBE GENERAL AREAS TO BE CONSIDERED
1
PROJECT SPECIFICATION
Scope & Limitations
General Area(s) to be studied
PROJECT MANAGER OR CHIEF CONTACT
2
Chief Contact(name)
Address
Phone
List Specific Functions to be Covered in Analysis:
LIST ALL MAJOR ITEMS TO BE COVERED BY THE PROJECT
3
Describe Specific Project Constraints/Limitations:
DESCRIBE SPECIFIC ITEMS WHICH WILL CONSTRAIN OR LIMIT ANY PART OF THE PROJECT
4

FIGURE - 3

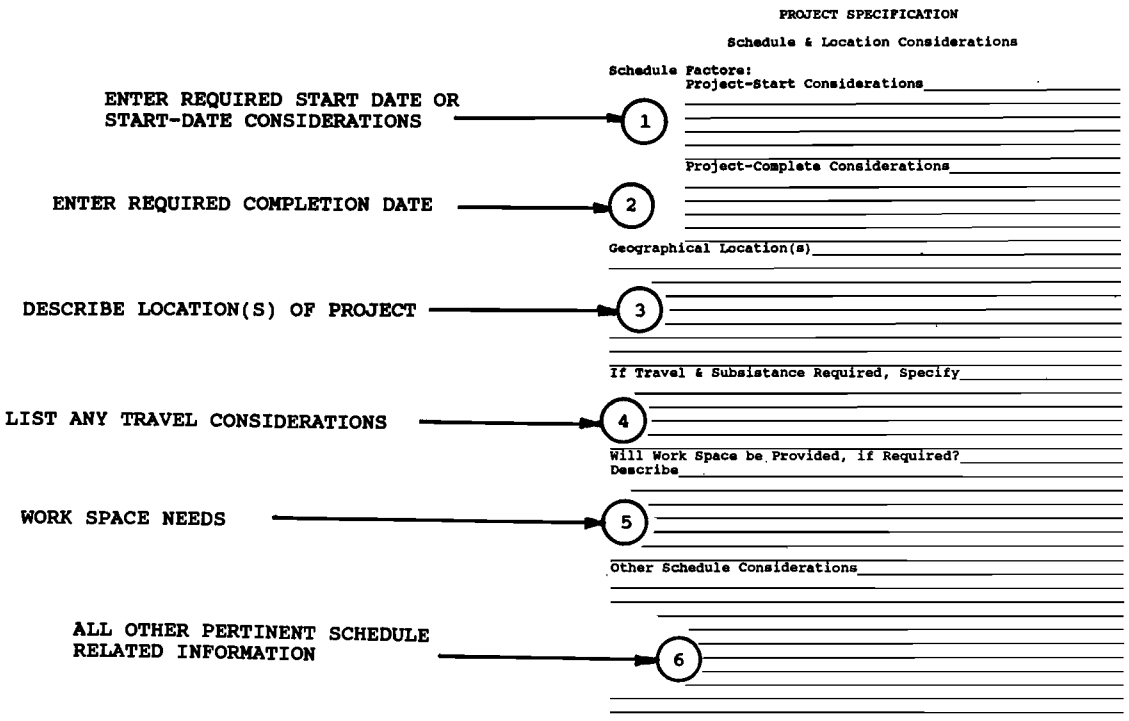

PROJECT SPECIFICATION
Schedule & Location Considerations
Schedule Factors:
Project-Start Considerations
Project-Complete Considerations
Geographical Location(s)
If Travel & Subsistance Required, Specify
Will Work Space be Provided, if Required?
Describe
Other Schedule Considerations
ENTER REQUIRED START DATE OR START-DATE CONSIDERATIONS
1
ENTER REQUIRED COMPLETION DATE
2
DESCRIBE LOCATION(S) OF PROJECT
3
LIST ANY TRAVEL CONSIDERATIONS
4
WORK SPACE NEEDS
5
ALL OTHER PERTINENT SCHEDULE RELATED INFORMATION
6

FIGURE - 4

TASK PLANNING

The Task Planning worksheet is one of the most important project management tools, since it is used in breaking down the tasks required to do a job.

We offer two versions of the Task Planning sheet; the Task Planning worksheet and the Task Assignment Matrix.

The first consideration in task planning is that each task must have a separate and unique control number. Once assigned, these numbers will carry through and serve as a cross-reference for tasks in all other documents used in your project. Any numbering scheme that is convenient for you may be applied. However, it is important that it should be such a scheme as to allow for later addition or deletion of sub-tasks without the necessity for renumbering.

The next important aspect in task planning is that each task title denotes an actual element of work. There is no hard and fast rule as to the level to which tasks should be broken down. Rather they should be defined only to that level where the total task can be easily understood and resources (such as manhours) accurately estimated.

Figure - 5 illustrates how to use the basic Task Planning worksheet.

Figure - 6, Task Assignment Matrix, provides for an optional method for assigning tasks by major groups and/or to specific task leaders.

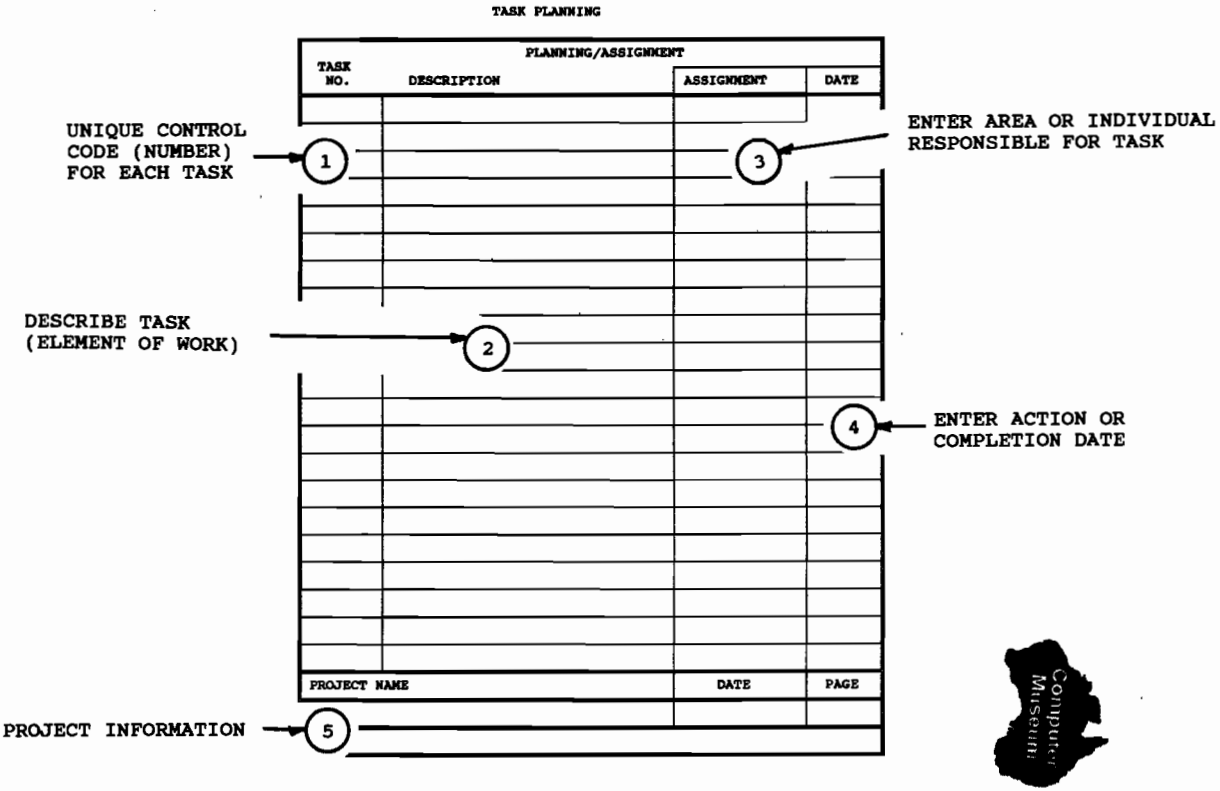

TASK PLANNING
PLANNING/ASSIGNMENT
TASK NO.
DESCRIPTION
ASSIGNMENT
DATE
UNIQUE CONTROL CODE (NUMBER) FOR EACH TASK
1
ENTER AREA OR INDIVIDUAL RESPONSIBLE FOR TASK
3
DESCRIBE TASK (ELEMENT OF WORK)
2
4
ENTER ACTION OR COMPLETION DATE
PROJECT NAME
DATE
PAGE
PROJECT INFORMATION
5

FIGURE - 5

TASK PLANNING(ASSIGNMENT MATRIX)

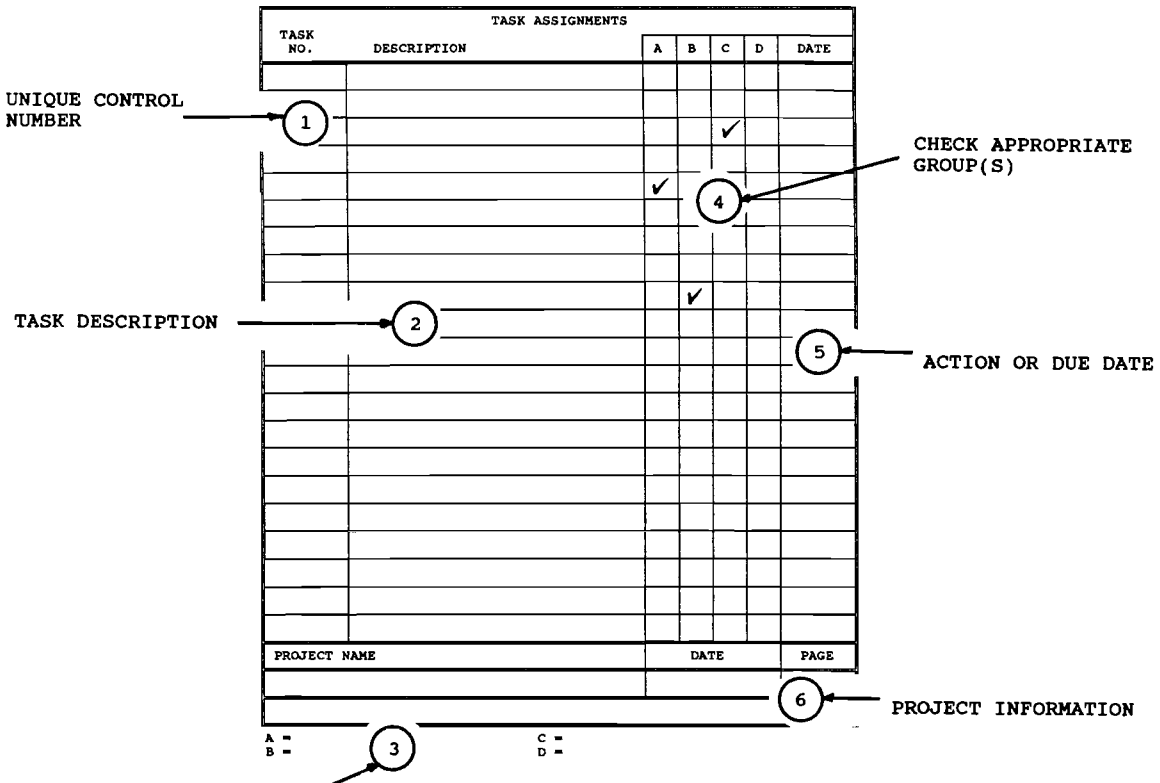
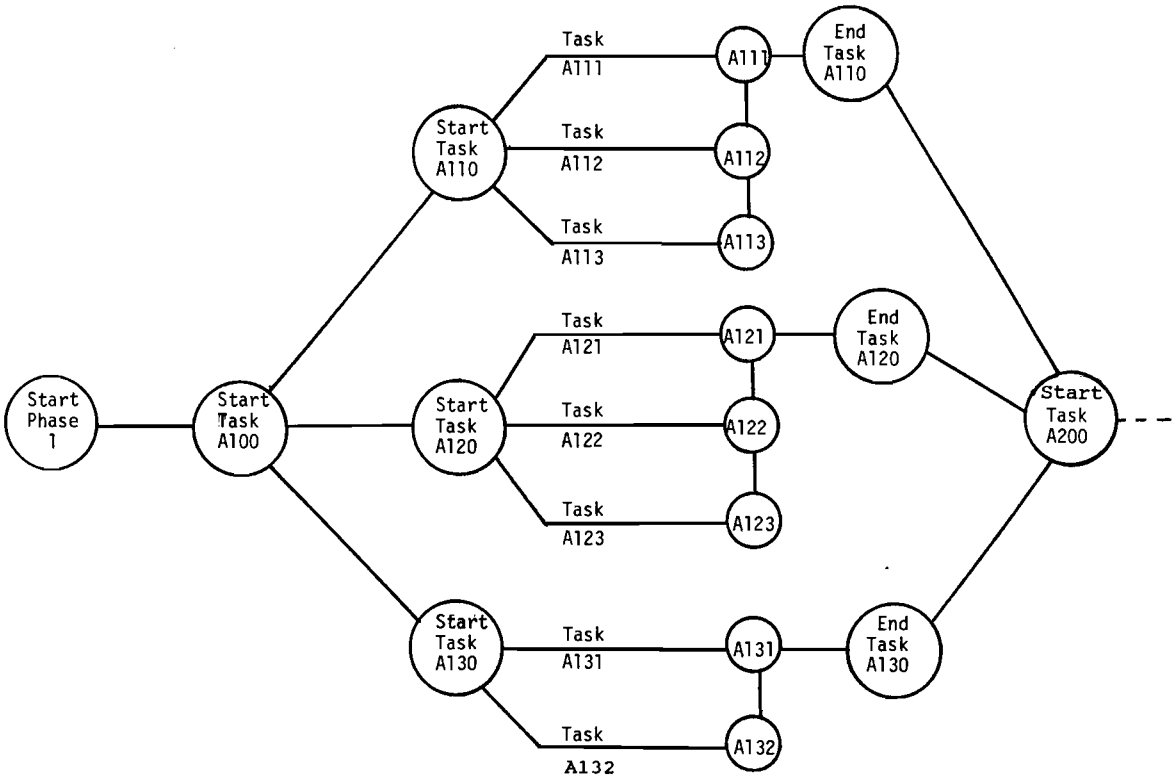


FIGURE - 6

NETWORK DEVELOPMENT

Once the tasks have been satisfactorily defined, the next important planning effort involves determining the interdependencies and sequence of the tasks. A useful technique for this is the network. In network terminology, circles represent events (start, stop, etc.) while lines indicate activities (tasks).

Figure - 7 shows a portion of a typical network which displays the sequence inwhich a group of tasks should be performed.



Network Development - Figure-7

MILESTONE SCHEDULES (GANTT CHARTS)

The project network plan must now be translated into some mode suitable for control purposes. The technique most often used is a Milestone Chart which utilizes key tasks extracted from the network.

We have provided a typical example of a milestone chart, Figure - 8, with applicable descriptions of its use.

Again you see task number and description which is related to a time line to show task progress as a function of time. We show our chart in weeks, however, almost any unit of time (hours, days, weeks, etc.) may be used as dictated by your specific project needs.

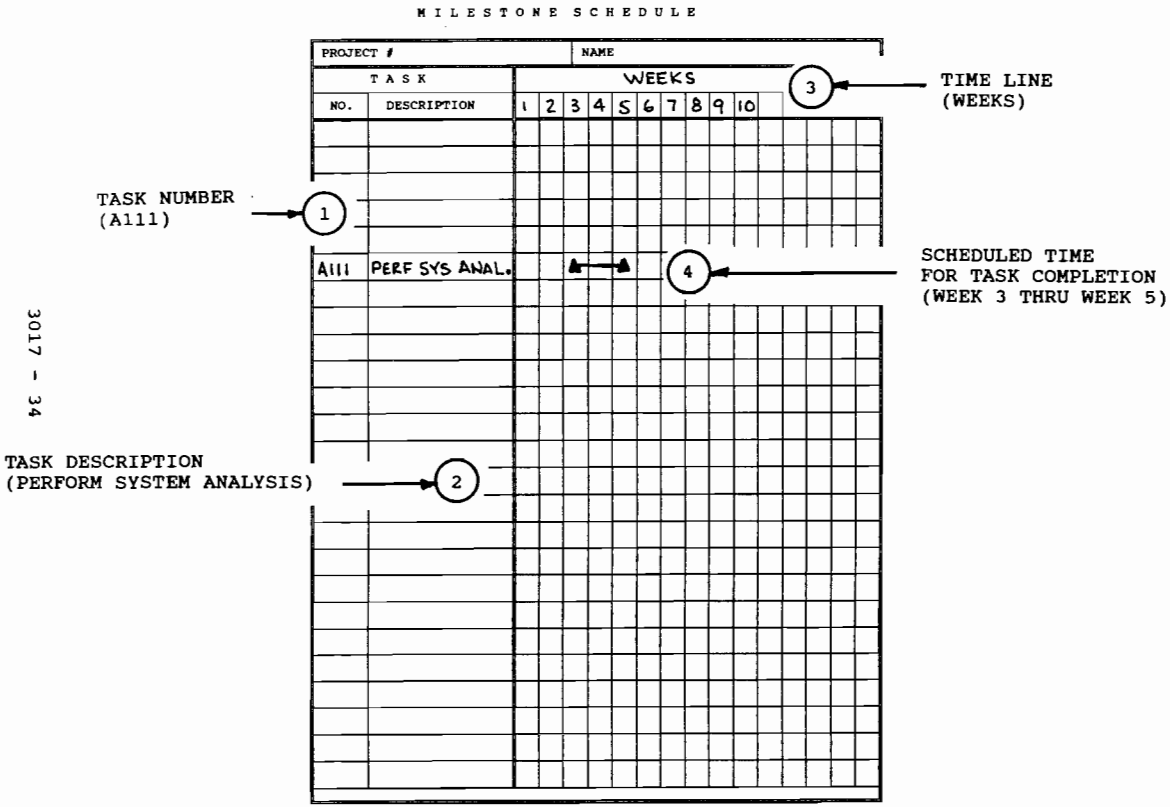

MILESTONE SCHEDULE
PROJECT #
NAME
TASK
NO.
DESCRIPTION
WEEKS
1 2 3 4 5 6 7 8 9 10
A111
PERF SYS ANAL.
TASK NUMBER
(A111)
1
TASK DESCRIPTION
(PERFORM SYSTEM ANALYSIS)
2
3
TIME LINE
(WEEKS)
4
SCHEDULED TIME
FOR TASK COMPLETION
(WEEK 3 THRU WEEK 5)

FIGURE - 8

RESOURCE PLANNING

Once the network tasks have been defined and sequenced, the job of estimating manpower and material requirements for the project is the next step. To assist in this step we provide the Resource Planning worksheet, Figure - 9. This sheet is specifically designed to record manhours for upto four resource levels (ie. PA1, PA2, PA3, System Analyst), however, with minor changes in the resource worksheet we could record any material or piece of equipment needed to meet a particular project objective.

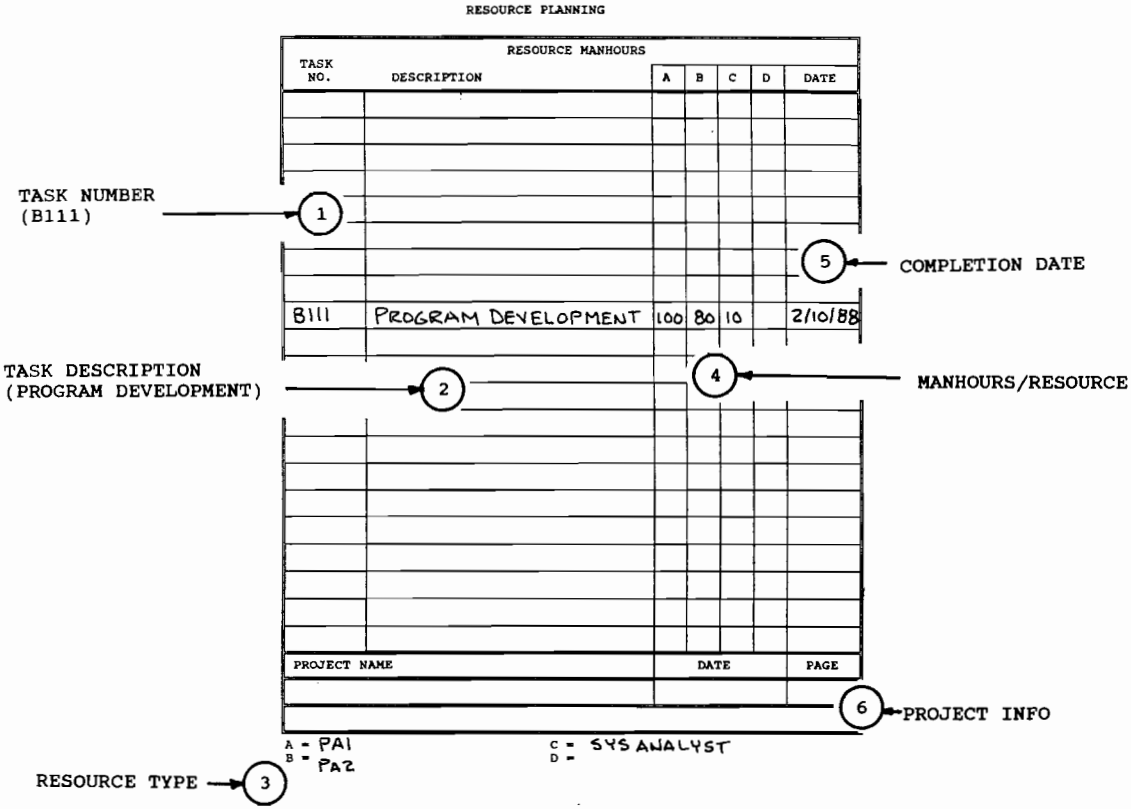

RESOURCE PLANNING
RESOURCE MANHOURS
TASK NO.
DESCRIPTION
A
B
C
D
DATE
B111
PROGRAM DEVELOPMENT
100
80
10
2/10/88
PROJECT NAME
DATE
PAGE
TASK NUMBER (B111)
1
TASK DESCRIPTION (PROGRAM DEVELOPMENT)
2
RESOURCE TYPE
3
4
MANHOURS/RESOURCE
5
COMPLETION DATE
6
PROJECT INFO
A = PA1
B = PA2
C = SYS ANALYST
D =

FIGURE - 9

IMPLEMENTATION AND CONTROL

The Implementation and control phase of your project requires your undivided attention, since this is where you reap the benefits of the initial planning.

Communications is keen to this phase and must be continuous and well documented.

Over the years we have developed many worksheets/forms which will help you implement and control your project better and more efficiently:

Figure - 10	Resource Record
Figure - 11	Project Review Form
Figure - 12	Bill of Materials
Figure - 13	Letter of Transmittal
Figure - 14	Project Communications Log
Figure - 15	Project Notes
Figure - 16	Project Meeting Notes

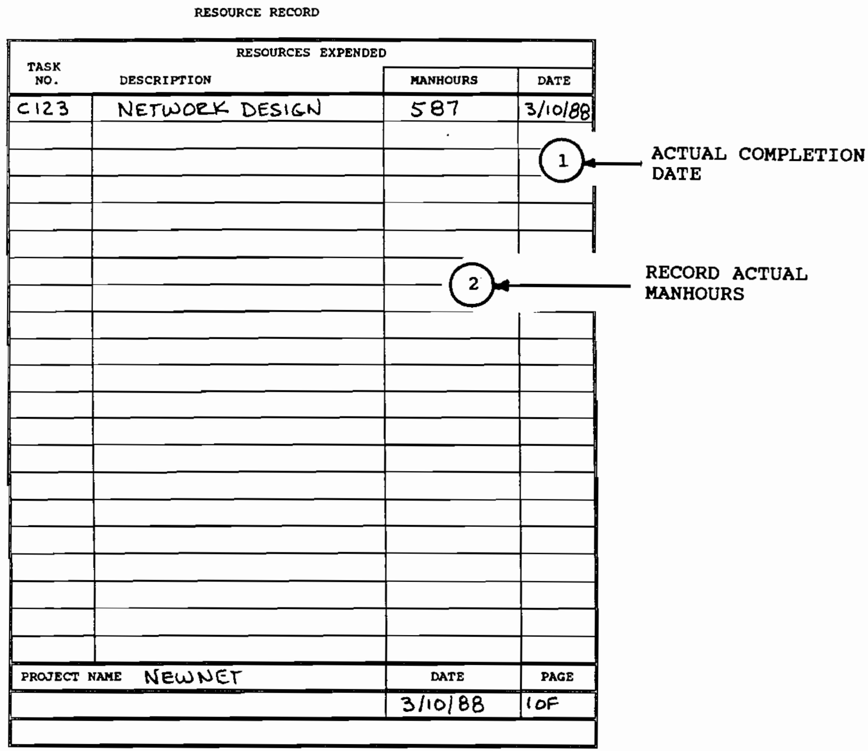
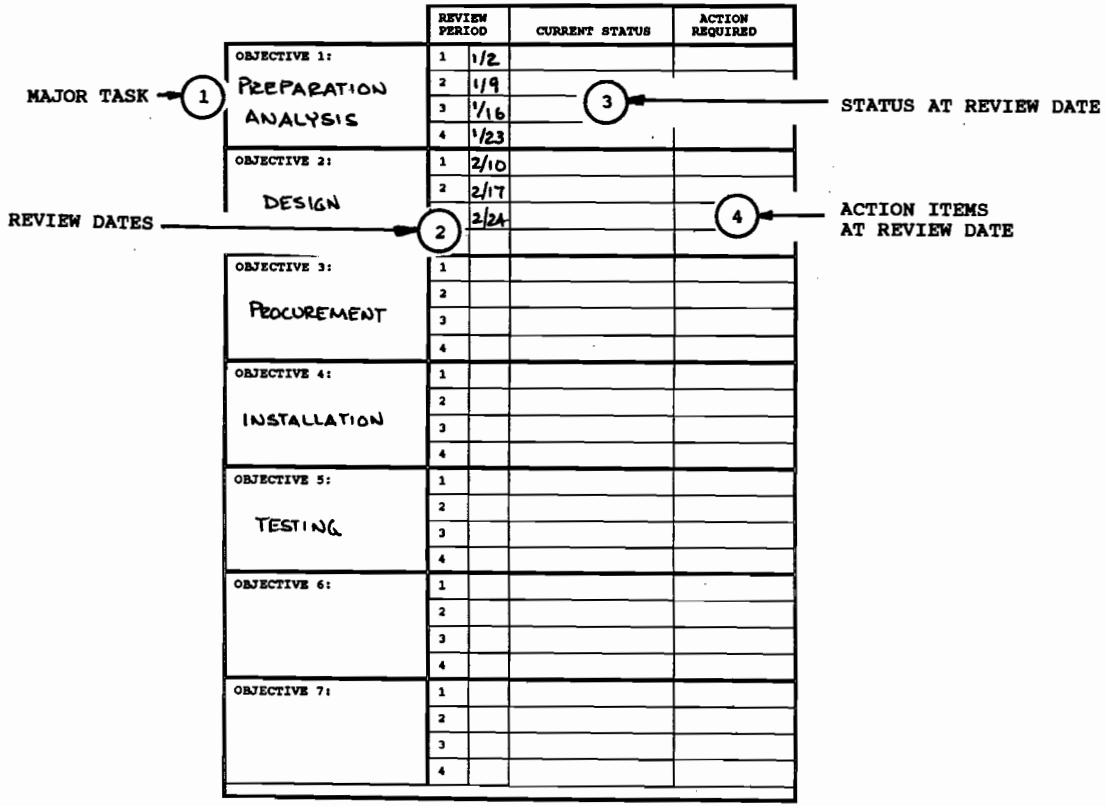

RESOURCE RECORD
RESOURCES EXPENDED
TASK NO.
DESCRIPTION
MANHOURS
DATE
C123
NETWORK DESIGN
587
3/10/88
1
ACTUAL COMPLETION DATE
2
RECORD ACTUAL MANHOURS
PROJECT NAME
NEWNET
DATE
PAGE
3/10/88
1 OF

FIGURE - 10

PROJECT REVIEW LOG



	REVIEW PERIOD		CURRENT STATUS	ACTION REQUIRED
OBJECTIVE 1: PREPARATION ANALYSIS	1	1/2		
	2	1/9		
	3	1/16		
	4	1/23		
OBJECTIVE 2: DESIGN	1	2/10		
	2	2/17		
	3	2/24		
	4			
OBJECTIVE 3: PROCUREMENT	1			
	2			
	3			
	4			
OBJECTIVE 4: INSTALLATION	1			
	2			
	3			
	4			
OBJECTIVE 5: TESTING	1			
	2			
	3			
	4			
OBJECTIVE 6:	1			
	2			
	3			
	4			
OBJECTIVE 7:	1			
	2			
	3			
	4			

BILL OF MATERIALS

ITEM	QUANTITY	DESCRIPTION	U/PRICE	EXT
1	1	HP3000/925LX	87,850	87,850

PROJECT NAME: | DATE: | PAGE OF

FIGURE - 12

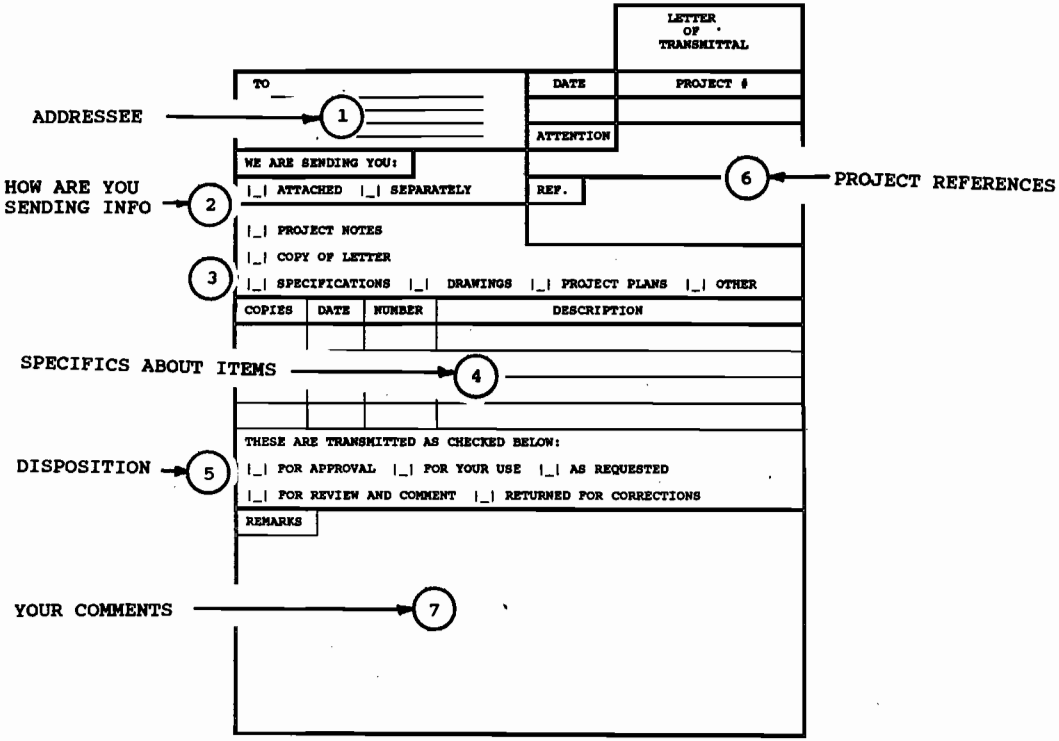

LETTER OF TRANSMITTAL
TO
DATE
PROJECT #
ATTENTION
ADDRESSEE
1
WE ARE SENDING YOU:
HOW ARE YOU SENDING INFO
2
|_| ATTACHED |_| SEPARATELY
REF.
6
PROJECT REFERENCES
|_| PROJECT NOTES
|_| COPY OF LETTER
3
|_| SPECIFICATIONS |_| DRAWINGS |_| PROJECT PLANS |_| OTHER
COPIES
DATE
NUMBER
DESCRIPTION
SPECIFICS ABOUT ITEMS
4
THESE ARE TRANSMITTED AS CHECKED BELOW:
DISPOSITION
5
|_| FOR APPROVAL |_| FOR YOUR USE |_| AS REQUESTED
|_| FOR REVIEW AND COMMENT |_| RETURNED FOR CORRECTIONS
REMARKS
YOUR COMMENTS
7

FIGURE - 13

PROJECT COMMUNICATIONS LOG		
PROJECT -		
DATE	INDIVIDUAL(S)	NOTES

FIGURE - 14

PROJECT NOTES

DATE	NAME	REF. NAME	PROJECT

SUBJECT	REFERENCE

ITEM	NOTES

PAGE OF

FIGURE - 15

PROJECT MEETING NOTES

DATE	NAME	PROJECT#

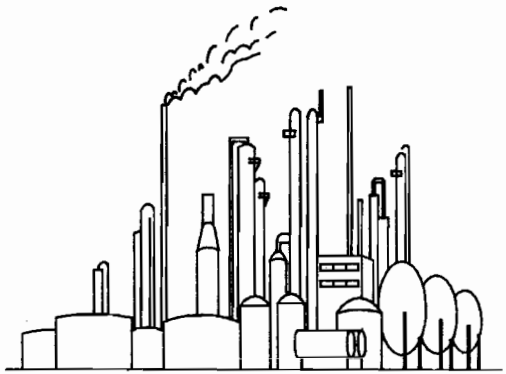
SUBJECT	REFERENCE

NOTES

	PAGE OF

FIGURE - 16

PART 3

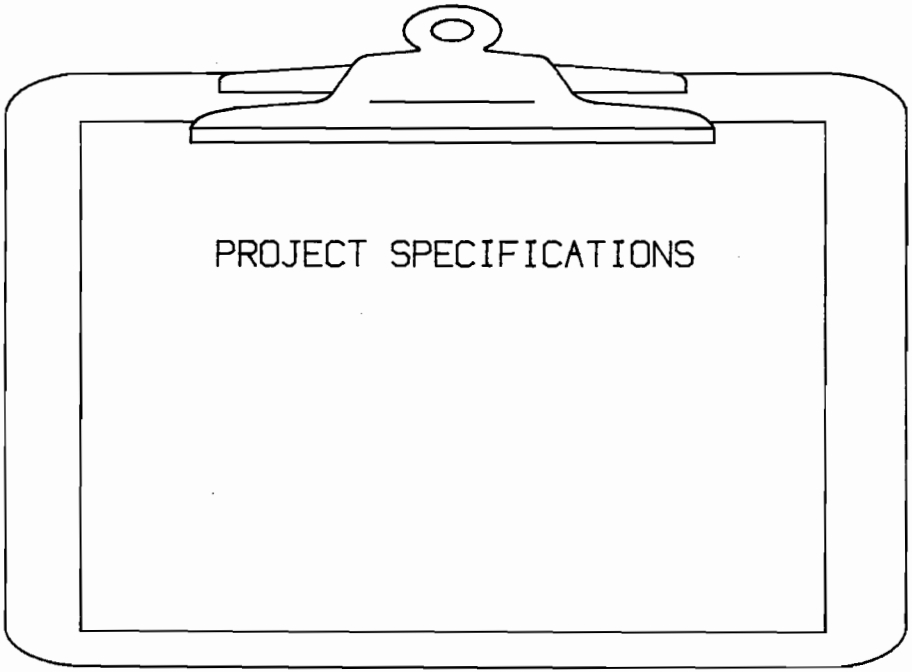


DEVELOPING A PROJECT

DEVELOPING A PROJECT

* WRITING THE SPECIFICATION
* PLANNING THE PROJECT
* CREATING THE TASK LIST
* DEVELOPING THE NETWORK DIAGRAM
* SELECTING THE PROJECT TEAM
* ASSIGNING THE TASKS
* CREATING THE SCHEDULE(S)
* DETERMINING THE RESOURCE REQUIREMENTS
* IMPLEMENTING AND CONTROLLING THE PROJECT

DEVELOPING A PROJECT



FOLLOWING THE STEPS

PROJECT SPECIFICATION

We will begin by creating a general System Specification using our System Specification Sheet discussed in Part 2 of our session.

As you will see, I have already filled in the very basic requirements of our "Online Tracking System".

From information gathered in creating the System Specification, certain Project Specification must also be determined. These include a definition of the functions that are to be covered in the process/analysis, and a determination of whether there are any project constraints or limitations.

Using our Project Specification Form, we can enter information unique to our project.

I have also included one additional form (Objectives Considerations Checklist) sometimes useful in determining objectives when justifying a particular project.

SYSTEM SPECIFICATION

Name of System__Online Problem Tracking System

List Current Problem(s)

1) Present batch system to slow
2) Inadequate Reports
3) No online access to data

List Prime Objective(s)/Goal(s)

Provide an interactive (online) problem tracking system for all ISD related user problems (ie Telephone, Datacomm, System, Application, etc.)

Describe Specific System Constraints/Limitations

Must operate on existing system/peripheral hardware.

Briefly Describe Existing DP/COMM Equipment

HP3000/58, 40 terminals, 5 printers, 3-571Mb disc drives, 1 6250BPI tape drive, 2 - 8 channel muxes.

Is Current System Documented Yes
Is Documentation Available Yes
Comments:

PROJECT SPECIFICATION

Scope & Limitations

General Area(s) to be studied__System/Program Development___
and installation._____
Chief Contact(name)____Joe Smith_____
Address___Mesa, AZ_____ Phone___991-1234_____
List Specific Functions to be Covered in Analysis:_____

_____
_____Macro calls_____
_____Copy Library_____
_____Data Base(s)/files_____
_____Screens_____
_____Reports/formats_____
_____User preferences_____
_____Maintenance/report programs_____
_____JCL_____

Describe Specific Project Constraints/Limitations:_____

_____
_____All programs must be interactive and must____
_____function on existing hardware._____

PROJECT SPECIFICATION

Schedule & Location Considerations

Schedule Factors:

Project-Start Considerations

Start time can occur anytime after Jan 15, 1988.

Project-Complete Considerations

Completion should be no later than November 15, 1988.

Geographical Location(s)

Local ISD

If Travel & Subsistance Required, Specify

Special training (2 weeks) at California location

Will Work Space be Provided, if Required?

Describe

2 workstations at user location for user training.

Other Schedule Considerations

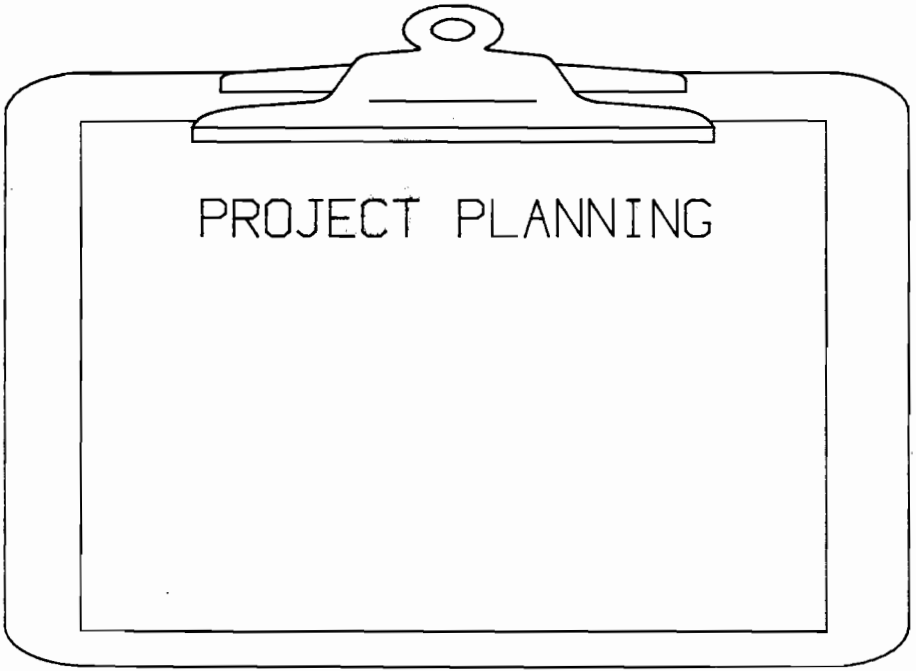
Additional contracting/consulting is authorized to meet overall schedule.

OBJECTIVES CONSIDERATIONS

CHECKLIST

	CHK	PRIORITY
REDUCE WASTE (Efficiency)		
Descrease Workload		
Reduce Overtime		
Free Skilled People's Time		
Reduce "Down or Off" Time of Equipment		
Increase Off-line Operations		
REDUCE COST (Economy)		
Reduce Number of Operations Performed		
Reduce Size or Qty of Equipment		
Reduce Manhour Requirements		
Reduce Number of Reports Produced		
Reduce Distributions of Reports		
BETTER INFORMATION (Accuracy)		
Increase Accuracy of Input/Output	✓	
Select Better Reporting Elements		
Improve Distribution	✓	
Increase Capacity of Equipment		
Increase Flexibility of Processes	✓	
MORE CURRENT INFORMATION (Timeliness)		
Decrease Throughput Time	✓	
Cut-Down Process Turnaround Time	✓	
Reduce Reproduction Service Time		
Reduce Transmission Times		
Provide More Frequent Reports	✓	
MORE OUTPUT FROM AVAILABLE RESOURCES (Productivity)		
Increase Hardware Utilization		
Provide More Compact Storage		
Provide Greater Compaction of Data		
Eliminate Redundant Operations	✓	
Eliminate Unnecessary Operations	✓	

DEVELOPING A PROJECT



FOLLOWING THE STEPS

PROJECT PLAN

I guess the better way to state this is really, **PLANNING THE PROJECT**.....

Project Management is the process by which it is assured that the objective is achieved and resources are not wasted. Planning is one of the two essential parts of project management. Control is the other.

Probably the most neglected area in design/development involves planning and control of the project. More than one diastrous project has been launched by computer people who communicated their aims to unknowing users using DP 'buzz words' in lieu of specific lists of easily understood tasks, schedules, and costs.

Any project must first be planned in detail. Control is involved with comparing actual progress with the actual plan and taking corrective action when the two do not correspond. Without the plan, true control is not possible; the need for corrective action, its nature, extent, and urgency cannot accurately be determined.

The Project Plan must include the following:

* Definition of the objective(s) or product goal
* Definition and description of tasks needed to be performed in order to meet the objective(s)
* Determining interdependencies and interrelationships of various tasks, and the scheduling of these tasks
* Determing the resources required to the tasks
* Logically grouping the tasks and assigning them
* Pricing and budgeting for all tasks

PROJECT PLAN LIST

DEFINITION OF THE OBJECTIVE(S)

DEFINITION AND DESCRIPTION OF TASKS NEEDED TO BE PERFORMED IN ORDER TO MEET THE OBJECTIVE(S).

DETERMINE INTERDEPENDENCIES AND INTERRELATIONSHIPS OF VARIOUS TASKS.

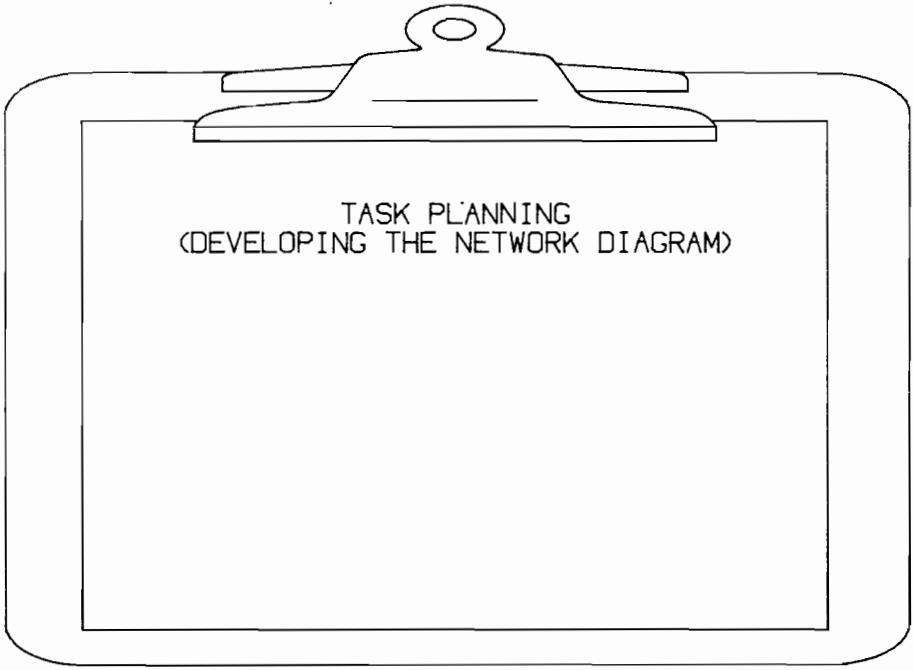
SCHEDULING ALL TASKS.

DETERMINE RESOURCES REQUIRED TO PERFORM ALL TASKS.

LOGICALLY GROUP AND ASSIGN TASKS

PRICE AND BUDGET FOR ALL TASKS

DEVELOPING A PROJECT



FOLLOWING THE STEPS

TASK PLANNING

There are several important considerations involved in breaking down tasks required to do a job.

The first consideration is that each task should have a unique control (task) number. Once assigned, these numbers will carry through and serve as a cross-reference for the same tasks in all project documents, including schedules, lists, assignments sheets, etc.

Any numbering scheme that is convenient and easy for the Project Manager or task leader may be applied. However, it should be a scheme as to allow for later additions or deletions of sub-tasks without the necessity of renumbering.

The next important aspect in task planning is that each task title denotes an actual element of work.

Once tasks have been broken down and defined, task statements (statements of work) can be generated.

With tasks satisfactorally defined, the next important planning effort involves determining the interdependencies and sequences of the tasks. A useful technique is the network. In network terminology, circles usually represent events, while lines represent tasks. We provide several examples.....

We will use the PERT (Program Evaluation & Review Technique) system to demonstrate the network.

There are three basic steps in creating and using the network:

1) Develop the network of events.

2) Estimate the time to accomplish each activity (task) in the network.

3) Analyze, through simple calculations, the schedule problems in getting from event to event.

TASK PLANNING

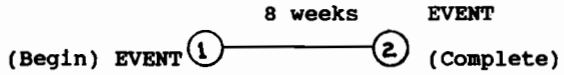
TASK NO.	PLANNING/ASSIGNMENT DESCRIPTION	ASSIGNMENT	DATE
1	CREATE PLAN		
2	PREP ANALYSIS		
3	DESIGN MACROS		
4	DESIGN DATA BASE		
5	DESIGN SCREENS		
6	DESIGN COPYLIB		
7	TEST MACROS		
	TEST COPYLIB		
	TEST SCREENS		
	TEST DATA BASE		
8	ASSIGN PROGRAMS		
9	DESIGN MAINT PROGS		
10	DESIGN REPORT PROGS		
11	DESIGN SPEC'L PROGS		
12	CODE MAINT PROGS		
13	CODE REPORT PROGS		
14	CODE SPEC'L PROGS		
15	TEST MAINT PROGS		
	TEST REPORT PROGS		
	TEST SPEC'L PROGS		

PROJECT NAME	DATE	PAGE
ONLINE PROBLEM TRACKING SYSTEM	1/20/88	1 OF

Step 1 - Rules

i) An event is a specific accomplishment required.

ii) The event must be recognizable as occurring at a specific point in time.



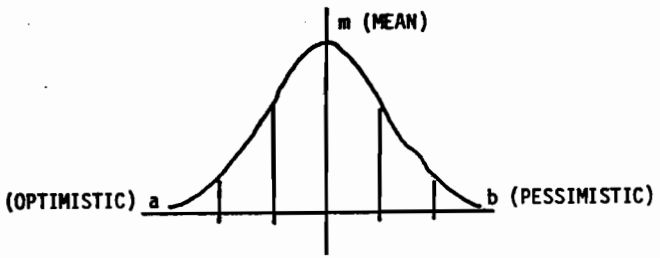
Step 2 - Rules

i) Define Optimistic Time (a)

ii) Define Pessimistic Time (b)

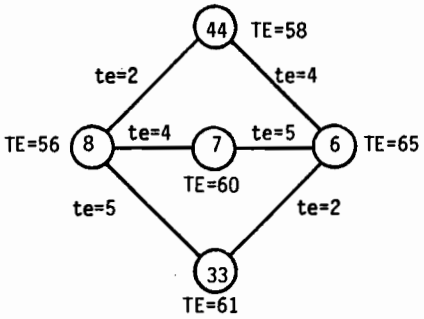
iii) Compute Most Likely Time (te)

$$te = \frac{a + 4m + b}{6}$$



Step 3 - Rules

i) Determine the amount of slack associated with various activities (this will define critical path).



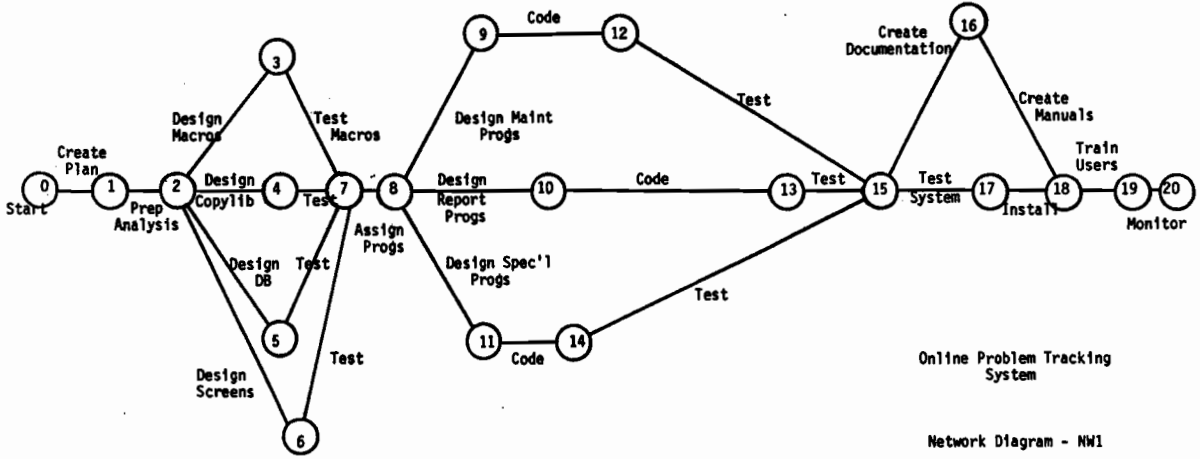
The longest time (56+4+5)=65

ii) Make resource/time adjustments.

From this information, we developed the basic network diagram of our project.

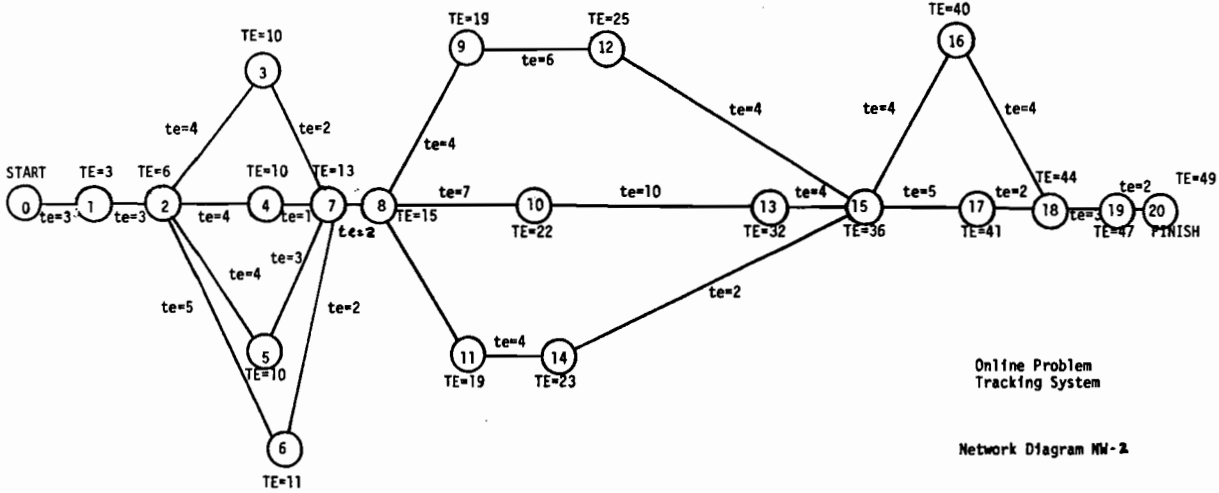
Figure NW-1 shows our network starting with event (1) and ending with event (20). As you can see, the network diagram describes each major event for our system project.

Figure NW-2 provides us with our estimated times to accomplish each task (te) and time (in weeks) when an event is started and/or completed (TE). As you can see, the major portion of our project was accomplished in about 49 weeks.



Online Problem Tracking System

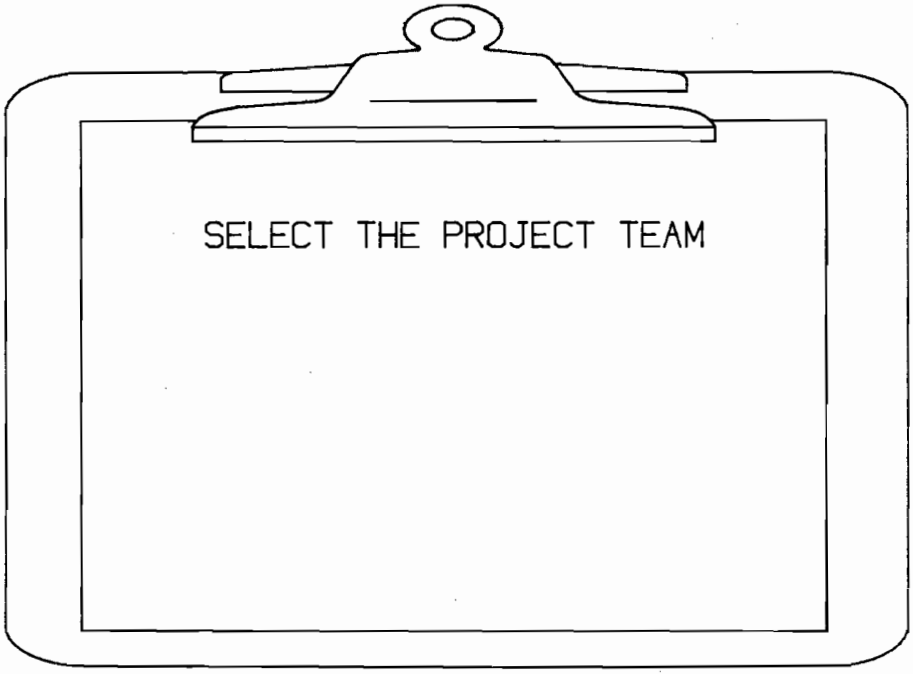
Network Diagram - NW1



Online Problem
Tracking System

Network Diagram NW-2

DEVELOPING A PROJECT



FOLLOWING THE STEPS

CREATE THE PROJECT TEAM

The creation of a project team can take place anytime before the actual task assignment and schedule step. Although there may be several levels of project participants, it is essential that we determine the appropriate task leaders/managers who will provide both guidance and local control of any group of interdependent tasks.

These key individuals are normally determined in the task planning step.

Typically the breakdown could include systems (hardware), software, communications, etc.

The project team includes all staff members who will contribute to the project.

The Project Manager sets the major task assignments for the project team, and both the Project Manager and task leaders assign tasks to other subordinate personnel.

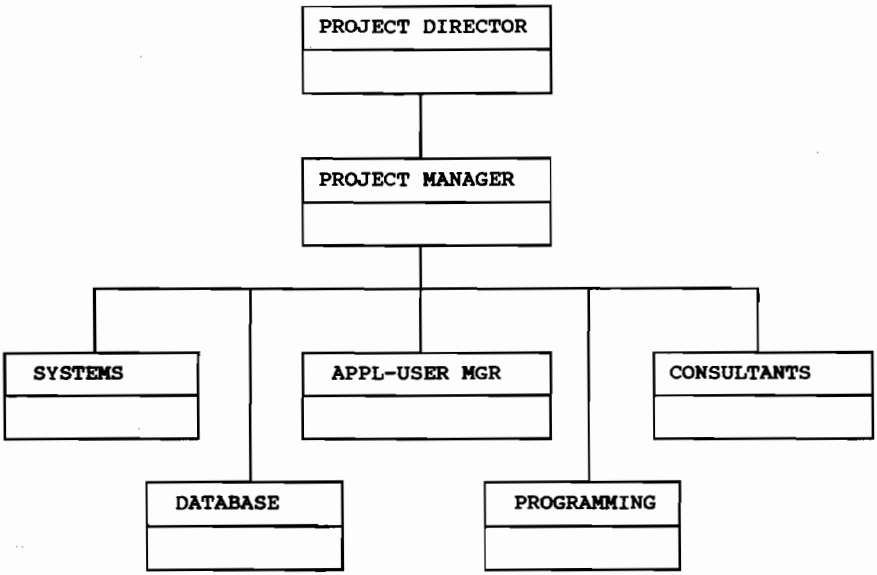
Project team members perform tasks according to a published project schedule and report their activities regularly.

Project members may work on the project full time or part time for either a short time or for the complete project.

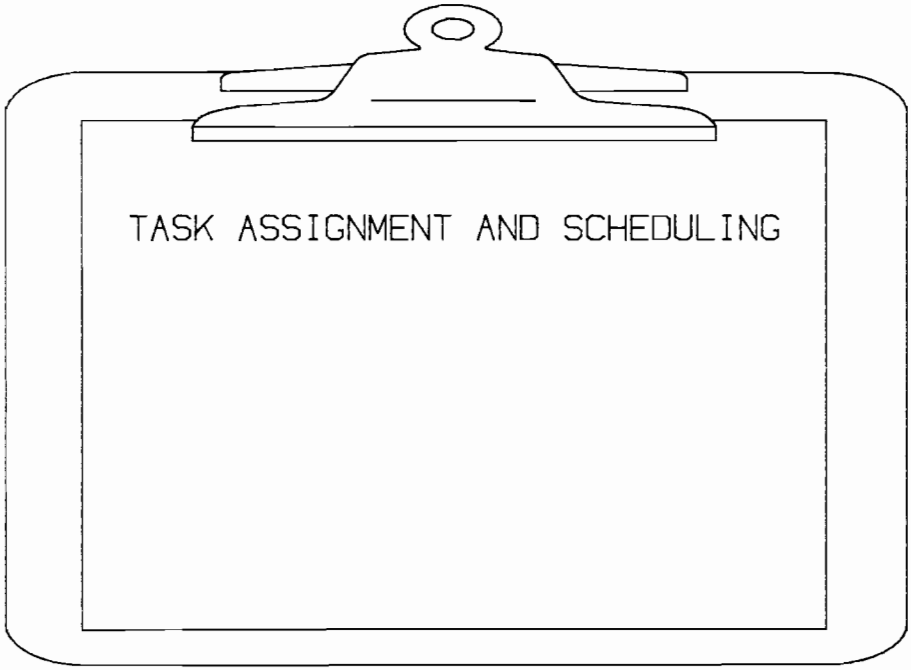
In some cases, contracting and/or consulting personnel can be employed to provide needed assistance.

Once a project begins, the Project Manager will keep the 'task leaders' informed of all changes in task requirements and schedules. This will allow the task leaders to coordinate the activities of their respective staff members.

ONLINE PROBLEM
TRACKING SYSTEM PROJECT
TEAM



DEVELOPING A PROJECT



FOLLOWING THE STEPS

ASSIGN/SCHEDULE THE TASKS

Once the network is complete and the project team is in place, task assignment and control schedules can be developed.

Input for schedules include the network, project specifications, and task planning data. The scheduling of tasks should normally be done in conjunction with the planning of resources.

The process may take several sub-steps in order to provide the best estimate of time/resources to accomplish all of the assigned tasks.

* **ASSIGNING THE TASKS**

  We now are ready to assign our MAJOR TASKS, which we determined in the Task Planning phase.

  We use the Task Planning (Assignment Matrix) sheet to enter our assignments.

  Each MAJOR TASK (1 - 15) will be managed by the respective (A thru D) project leader. Further sub-divisions (assignments) are the responsibility of each project leader.

  To further allow you to organize your project, you may set up Major Tasks/Groups.

  These task groups define major areas of project responsibility and management.

* Milestone Schedule(s)

  The milestone schedule (GANTT CHART) we offer at this time includes only major tasks and extends for 49 weeks. This time window has been selected since it represents the best estimate for the overall project completion.

TASK PLANNING(ASSIGNMENT MATRIX)

TASK NO.	DESCRIPTION	TASK ASSIGNMENTS A	B	C	D	DATE
1	CREATE PLAN					
2	PREP ANALYSIS	X				
3	DESIGN MACROS			X		
4	DESIGN DATA BASE		X			
5	DESIGN SCREENS			X		
6	DESIGN COPYLIB			X		
7	TEST MACROS			X		
	TEST COPYLIB			X		
	TEST DATA BASE		X	X		
8	ASSIGN PROGRAMS	X				
9	DESIGN MAINT PROGS			X		
10	DESIGN REPORT PROGS			X		
11	DESIGN SPEC'L PROGS			X		
12	CODE MAINT PROGS				X	
13	CODE REPORT PROGS				X	
14	CODE SPEC'L PROGS				X	
15	TEST MAINT PROGS			X	X	
	TEST REPORT PROGS			X	X	
	TEST SPEC'L PROGS			X	X	

PROJECT NAME	DATE	PAGE
ONLINE PROBLEM TRACKING SYSTEM	1/20/88	1 OF

A = SYS ANALYST
B = DATABASE ADMIN
C = PROG/ANALYST
D = PROGRAMMER

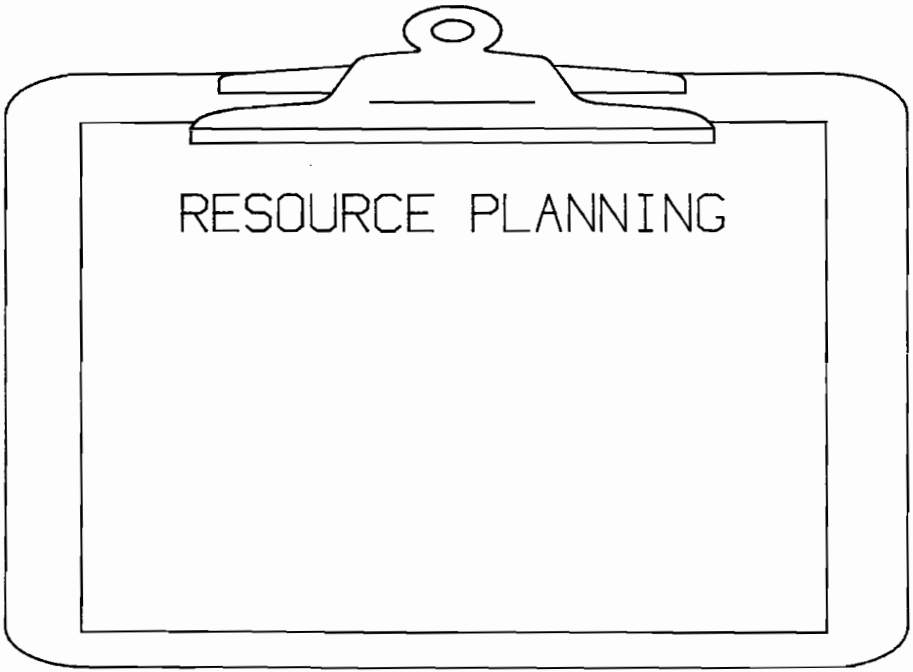
Obviously, this schedule will be revised by actual project results and time.

Additional milestone schedules of greater detail and higher resolution (daily) can be kept by each project leader.

M I L E S T O N E S C H E D U L E

PROJECT # 1234	NAME ONLINE PROBLEM TRACKING SYS																	
TASK DESCRIPTION	JAN			FEB			MAR				APR					MAY		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
CREATE PLAN	>>>>>>>>																	
PREP ANALYSIS				>>>>>>>>														
DESIGN MACROS							>>>>>>>>>>>											
DESIGN DB							>>>>>>>>>>>											
DESIGN SCRNS							>>>>>>>>>>>											
DESIGN COPYLB							>>>>>>>>>>>											
TEST MACROS											>>>>>							
TEST COPYLB											>>							
TEST SCRNS											>>>>>							
TEST DB											>>>>>>>>							
ASSIGN PROGS														>>>>>				
DESIGN MAINT P																>>>>>>>>		
DESIGN RPT P																>>>>>>>>		
DESIGN SPCL P																>>>>>>>>		
CODE MAINT P																		
CODE RPT P																		
CODE SPCL P																		
TEST MAINT P																		
TEST RPT P																		
TEST SPCL P																		

DEVELOPING A PROJECT



FOLLOWING THE STEPS

RESOURCE PLANNING

Once the network tasks have been defined and sequenced, the job of estimating manpower and material requirements for the project is the next order of business.

Manpower and other resources required to accomplish the defined implementation tasks are developed with the Resource Planning worksheet. The format is similar to the Task Planning sheet.

In using the Resource Estimating worksheet, manpower estimates can be identified in terms of individuals or in terms of skill types.

We elected to record resources by major task groups (A thru D) and in manhours of time.

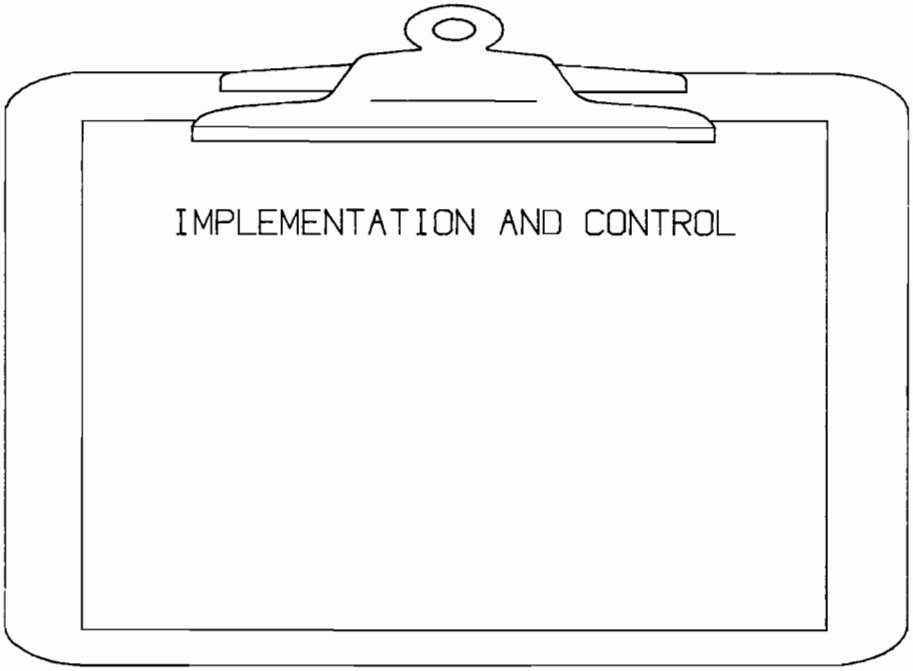
RESOURCE PLANNING

TASK NO.	DESCRIPTION	RESOURCE MANHOURS A	B	C	D	DATE
1	CREATE PLAN					
2	PREP ANALYSIS	240				
3	DESIGN MACROS			160		
4	DESIGN DATABASE		160			
5	DESIGN SCREENS			240		
6	DESIGN COPYLIB			160		
7	TEST MACROS			80		
	TEST COPYLIB			40		
	TEST SCREENS			80		
	TEST DATA BASE		120			
8	ASSIGN PROGRAMS	80				
9	DESIGN MAINT PROGS			320		
10	DESIGN REPORT PROGS			280		
11	DESIGN SPEC'L PROGS			280		
12	CODE MAINT PROGS				240	
13	CODE REPORT PROGS				800	
14	CODE SPEC'L PROGS				160	
15	TEST MAINT PROGS			160		
	TEST REPORT PROGS			320		
	TEST SPEC'L PROGS			100		

PROJECT NAME	DATE	PAGE
ONLINE PROBLEM TRACKING SYSTEM		

A = SYS ANALYST
B = DB MGR
C = PROG/ANALYST
D = PROG

DEVELOPING A PROJECT



FOLLOWING THE STEPS

IMPLEMENTATION AND CONTROL

Regardless of the planning excellence, there is one overriding consideration. The credibility of the entire plan, and the task leaders ability to perform to that plan, will be judged by managements' observance of the task leaders performance during the earlier analysis and design portion of a project.

It is, therefore, important that continuous feedback and feedforward information be supplied to control the overall project.

* Establish regular intervals (typically monthly) for overall project meetings with your project team, providing updated and field input as to progress/change of the project.

* Have individual meetings with each task leader (weekly) giving special attention to detail/progress of each task in the respective group.

* Provide regular updated documentation to all members, which includes revised task lists/descriptions, GANTT charts indicating critical path, and overall PERT charts of the project.

* Be alert to any and all problems that arise, and make quick decisive corrections whenever necessary.

* Resolve any conflicts immediately.

* Stick to the plan/schedule.

	REVIEW PERIOD		CURRENT STATUS	ACTION REQUIRED
OBJECTIVE 1: PREPARATION/ ANALYSIS	1			
	2			
	3			
	4			
OBJECTIVE 2: DESIGN	1			
	2			
	3			
	4			
OBJECTIVE 3: PROGRAM	1			
	2			
	3			
	4			
OBJECTIVE 4: TEST	1			
	2			
	3			
	4			
OBJECTIVE 5: INSTALLATION	1			
	2			
	3			
	4			
OBJECTIVE 6:	1			
	2			
	3			
	4			
OBJECTIVE 7:	1			
	2			
	3			
	4			

APPENDIX

* SAMPLE WORKSHEETS

SAMPLE WORKSHEETS

* System Specification
* Objectives Considerations Checklist
* Project Specification (Schedule & Location Considerations)
* Project Specification (Scope & Limitations)
* Project Review Log
* Task Planning (Assignment Matrix)
* Milestone Schedule (8 1/2 x 11)
* Milestone Schedule (11 x 8 1/2)
* Resource Planning
* Resource Record
* Project Notes
* Project Meeting Notes
* Project Communications Log
* Letter of Transmittal
* Bill of Materials

SYSTEM SPECIFICATION

Name of System_____

List Current Problem(s)_____

List Prime Objective(s)/Goal(s)_____

Describe Specific System Constraints/Limitations_____

Briefly Describe Existing DP/COMM Equipment_____

Is Current System Documented_____
Is Documentation Available_____
Comments:_____

OBJECTIVES CONSIDERATIONS

CHECKLIST

	CHK	PRIORITY
REDUCE WASTE (Efficiency)		
Descrease Workload		
Reduce Overtime		
Free Skilled People's Time		
Reduce "Down or Off" Time of Equipment		
Increase Off-line Operations		
REDUCE COST (Economy)		
Reduce Number of Operations Performed		
Reduce Size or Qty of Equipment		
Reduce Manhour Requirements		
Reduce Number of Reports Produced		
Reduce Distributions of Reports		
BETTER INFORMATION (Accuracy)		
Increase Accuracy of Input/Output		
Select Better Reporting Elements		
Improve Distribution		
Increase Capacity of Equipment		
Increase Flexibility of Processes		
MORE CURRENT INFORMATION (Timeliness)		
Decrease Throughput Time		
Cut-Down Process Turnaround Time		
Reduce Reproduction Service Time		
Reduce Transmission Times		
Provide More Frequent Reports		
MORE OUTPUT FROM AVAILABLE RESOURCES (Productivity)		
Increase Hardware Utilization		
Provide More Compact Storage		
Provide Greater Compaction of Data		
Eliminate Redundant Operations		
Eliminate Unnecessary Operations		

PROJECT SPECIFICATION

Schedule & Location Considerations

Schedule Factors:
Project-Start Considerations

Project-Complete Considerations

Geographical Location(s)

If Travel & Subsistance Required, Specify

Will Work Space be Provided, if Required?
Describe

Other Schedule Considerations

PROJECT SPECIFICATION

Scope & Limitations

General Area(s) to be studied_____

Chief Contact(name)_____

Address_____ Phone_____

List Specific Functions to be Covered in Analysis:_____

Describe Specific Project Constraints/Limitations:_____

	REVIEW PERIOD		CURRENT STATUS	ACTION REQUIRED
OBJECTIVE 1: PREP ANALYSIS	1			
	2			
	3			
	4			
OBJECTIVE 2: PRIMARY DESIGN	1			
	2			
	3			
	4			
OBJECTIVE 3: PROGRAM DESIGN	1			
	2			
	3			
	4			
OBJECTIVE 4: SYSTEM TEST	1			
	2			
	3			
	4			
OBJECTIVE 5: INSTALLATION	1			
	2			
	3			
	4			
OBJECTIVE 6:	1			
	2			
	3			
	4			
OBJECTIVE 7:	1			
	2			
	3			
	4			

TASK PLANNING(ASSIGNMENT MATRIX)

TASK NO.	DESCRIPTION	TASK ASSIGNMENTS A	B	C	D	DATE

PROJECT NAME	DATE	PAGE

A =
B =
C =
D =

M I L E S T O N E S C H E D U L E

PROJECT #	NAME

T A S K	
NO.	DESCRIPTION

PROJECT #	NAME

TASK NO.	TASK DESCRIPTION

RESOURCE PLANNING

TASK NO.	DESCRIPTION	RESOURCE MANHOURS A	B	C	D	DATE

PROJECT NAME	DATE	PAGE

A =
B =
C =
D =

RESOURCE RECORD

RESOURCES EXPENDED			
TASK NO.	DESCRIPTION	MANHOURS	DATE
PROJECT NAME		DATE	PAGE

PROJECT NOTES

DATE	NAME	REF. NAME	PROJECT

SUBJECT	REFERENCE

ITEM	NOTES

	PAGE OF

PROJECT MEETING NOTES

DATE	NAME	PROJECT#

SUBJECT	REFERENCE

NOTES

	PAGE OF

PROJECT COMMUNICATIONS LOG		
PROJECT -		
DATE	INDIVIDUAL(S)	NOTES

LETTER
OF
TRANSMITTAL

TO	DATE	PROJECT #
_____		
_____	ATTENTION	
_____		
_____		

WE ARE SENDING YOU:

|_| ATTACHED |_| SEPARATELY

REF.

|_| PROJECT NOTES

|_| COPY OF LETTER

|_| SPECIFICATIONS |_| DRAWINGS |_| PROJECT PLANS |_| OTHER

COPIES	DATE	NUMBER	DESCRIPTION

THESE ARE TRANSMITTED AS CHECKED BELOW:

|_| FOR APPROVAL |_| FOR YOUR USE |_| AS REQUESTED

|_| FOR REVIEW AND COMMENT |_| RETURNED FOR CORRECTIONS

REMARKS

B I L L   O F   M A T E R I A L S

ITEM	QUANTITY	DESCRIPTION	U/PRICE	EXT

PROJECT NAME:	DATE:	PAGE OF

TITLE: Getting the Most Out of Your Data Base

AUTHOR: Bradley Tashenberg (713) 621-2808
Bradmark Computer Systems
4265 San Felipe, Suite #820
Houston, TX 77027

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 3018

TITLE: Tutorial: Computer Security

AUTHOR: Isaac Blake (602) 731-8218
City of Tempe
120 East Fifth Street
Tempe, AZ 85281

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

(Handouts available at presentation.)

PAPER NO. 3019

Change for the Better

Tips for Managing Your Software Development and Maintenance Life Cycle

Daniel Cobb
Operations Control Systems
560 San Antonio Road
Palo Alto, California 94306
(415) 493-4122

Much of the emphasis on change control in management information systems departments has been initiated by concerns over the risks of computer fraud, malicious destruction of programs and data, and issues related to the reliability of audited financial data. However, good change control can make a major contribution to the goals of minimizing bugs and other production errors, and also maximizing the productivity of MIS staff. This paper looks at some of the common errors caused by poor change control, describes basic change control methods for preventing these errors, and explores some advanced change control concepts and solutions.

CHANGE CONTROL ERRORS

In our experience working with numerous MIS departments, over 50% of production problems are caused not by logic errors, but by various types of version control mistakes. The effects of these errors range from annoyance to catastrophe; nearly all could have been avoided by effective change control. Here are a few examples of version mistakes:

Fixing the Wrong Source

A program fails, leaving the sales department unable to enter orders. You are assigned to fix the problem, and, fortunately, it's easy to see what went wrong. You make a copy of the production source, fix the bug, test quickly, and move the new object into production. Users are elated. Two hours later your manager walks into your office. Your fix works fine, but the enhancement you implemented last month is gone! Apparently, you forgot to move the source back last month along with the new object when you implemented that change.

Fixing the Wrong Source, Part II

This time, you are assigned to make an enhancement to the inventory valuation system. Your shop allows unrestricted access to source code; programmers customarily make a backup copy of any source program before changing it. You make the backup copy, and begin work. You are called into an emergency meeting and asked to drop everything and work on a change to the general ledger interface which has to be completed before year-end, which is next week. Working under pressure, you are engrossed in the GL change, and forget about your half-made changes to the inventory programs. Meanwhile, an inventory valuation bug pops up. Another programmer is assigned to fix the program you had started to change. He backs up the source, fixes the bug, compiles, tests, and moves your partially completed, untested changes into production. Before the problem can be detected, numerous database records are updated incorrectly.

Concurrent Update

A programmer, John, is assigned to make a major change to a group of programs. He makes copies of all of them in his work group and begins work. Ten days later, a bug is discovered in one of the programs. A second programmer, Jean, is assigned to fix the bug. Jean, not realizing that John is already working on the program, makes her own copy, fixes the bug, tests the fix, and moves the program into production. John completes his major project several weeks later. He tests thoroughly, then moves the programs into production after coordinating database changes and production schedules. His enhancement works beautifully, but the bug Jean fixed is back.

Musical Versions

A software change requires changes to the database and to three programs that use it. You carefully build a test database and make copies of the three source files. You change the schema, make the structure changes to the test database, and modify the programs. You test thoroughly. Meanwhile your manager asks you to develop a report program using the new database structure. Finally, you move all four programs into production, and make the structure changes to the production database overnight. The next morning, it is apparent that something was forgotten: two other programs in the online system fail immediately. Major changes will be required to accommodate the new structure. Your only recourse is to roll back to the previous versions of the programs and restore the database from last night's backup. Fortunately, no changes have been made since the backup, other than your conversion. You restore the programs you changed from their backup copies - but one is missing. There is no AP1750O.PROD on disk. Someone forgot to rename the program before moving the new version into production, or renamed it to something you can't find...

These problems resulted from understandable human errors. But these errors could have been avoided by good change control procedures.

These are all examples of simple version control errors. More complex situations exist, such as the requirement to maintain multiple versions concurrently, or to maintain local changes to third-party software, and we will discuss some of these challenges later in this paper. But the majority of version control errors are of this simple variety, and can be avoided by basic change control procedure.

BASIC CHANGE CONTROL

Basic change control begins with some fundamental rules. The first of these is

RULE #1: Production source should be kept separate from test source, and programmers should not have the access to change it except through a controlled process.

As basic as this idea is, a substantial number of MIS departments, some of them quite large, do not follow it. Many others provide for the separation, but don't enforce it by restricting access (e.g., programmers have AM capability in the account where the production source is kept). This simple control will prevent "Fixing the Wrong Source, Part II", because the original source is never overwritten. The second programmer would make his own copy of the source from the original (of course, that could lead to concurrent update - see below).

The most common approach for keeping production and test files separate is called a "checkout-checkin" procedure. In the "checkout" phase, the programmer makes a copy of the production source in a test location, usually a separate account. After making changes and testing, the changed files are moved back into the production location, replacing the original source. The procedure is enforced by restricting programmers to read-only access to the production library. Someone other than the programmers typically performs the "checkin", or move-to-production step.

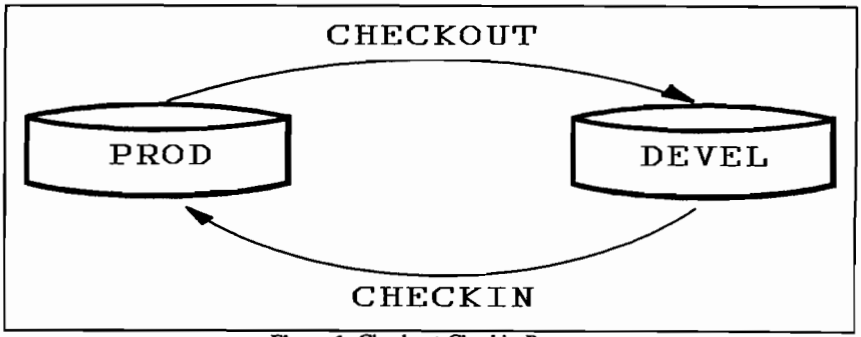


Figure 1: Checkout-Checkin Process

RULE #2: Establish a mechanism to prevent concurrent update by two or more programmers.

In many small shops, concurrent update is unlikely simply because each application is assigned to a single programmer. However, in any environment where there is a potential for two people to be working on the same program, it is a very real problem, and the bugs it can produce are some of the most insidious.

The most obvious method for preventing concurrent update is to establish some type of record of files checked out which is reviewed prior to each checkout operation. While this can be done manually, it is labor-intensive and error-prone, and therefore a candidate for automation. We have seen shops establish a physical documentation checkout in parallel with the files - the programmer signs out a program documentation binder before moving any files. If the binder isn't on the shelf, the program must be already out.

A third, more automated approach is to rename the files immediately after checkout, so that a subsequent attempt to copy the files will fail with a non-existent file error. This approach is simple and efficient, but it has three drawbacks: the person doing the checkout has to have write access to the production file to rename it, there's no way to find out who has the file if it's checked out, and the file is not found if you try to access it for any other reason (particularly a problem for mass moves). Finally, many shops prevent concurrent update by simply specifying the location in which development copies will reside and preventing overwrite through access restrictions or lockwords. If you go to check out a file and there's already a copy in the development location, you know it's already in use.

RULE #3: Keep your development area free of old copies of programs.

There are a number of ways to accomplish this, but basic diligence in limiting the source files in your development group to those being worked on is a good start. This also contributes to "Part II", above, and also could have prevented the failure to update the production source in the first example of "Fixing the Wrong Source". If it is exceptional for source program copies to reside in development groups, a neglected source program will stick out like a sore thumb whenever you do a LISTF.

RULE #4: Always make a backup copy of old software files before moving the new files to production. Use job streams or a software utility to reduce the opportunity for human error in this process.

A "skeleton" job stream can be used effectively to make sure that the old files are renamed or copied to another group, and that the files are all moved into production. Various techniques can be used to set a "flag" if any part of the move is unsuccessful. The job stream can also be set up to purge the files from the test area after a successful copy, which will help you to accomplish Rule #2, above. It might be worthwhile to write a simple utility to rename, copy, and purge a list of files. Commercial software is also available to do this.

RULE #5: Do not rely on memory to identify programs affected by changes to database structure or common code. Use a scan utility or maintain a "where-used" cross-reference.

Several utilities are commercially available for searching groups of files for a particular character string such as a dataset, item, or called routine name. These utilities are easy to use, and take the guesswork out of identifying the programs affected by a change to common code. As an alternative, you can maintain a manual or automated cross-reference of programs and common code used. This will be discussed in greater detail later in the paper.

ADVANCED CHANGE CONTROL TECHNIQUES

Concurrent Development and Maintenance

While most shops will never have the challenge of maintaining multiple concurrent versions as software vendors do, MIS departments are routinely called upon to make major, long term enhancements to applications while still maintaining the current, production system. Invariably, this raises the sticky issue of how to keep the development versions separate from the maintenance ones, and how to make sure the bug fixes and other maintenance changes made to the production system are integrated into the new version.

The first step in dealing with concurrent maintenance and development is to provide completely separate work areas for the two activities. While it may be tempting to avoid the overhead and inconvenience of working in separate accounts or groups, it is too easy to make mistakes when working with two or more files of the same or similar names in the same work area.

There are two workable approaches to forward integration of maintenance changes. One is to integrate each maintenance change into the development version immediately as soon as it is completed. This requires diligence on the part of the maintenance programmers, perhaps an additional check during the move-to-production process, and a good task list for the developers. The other approach is to make the copies for the development project at one time, take a "snapshot" backup of the source library at that time, then keep track of which programs have had maintenance changes since that time. Later in the project, use a maintenance log to identify the changes that have to be integrated, or use a source comparison utility to identify the particular code changes that have to be integrated.

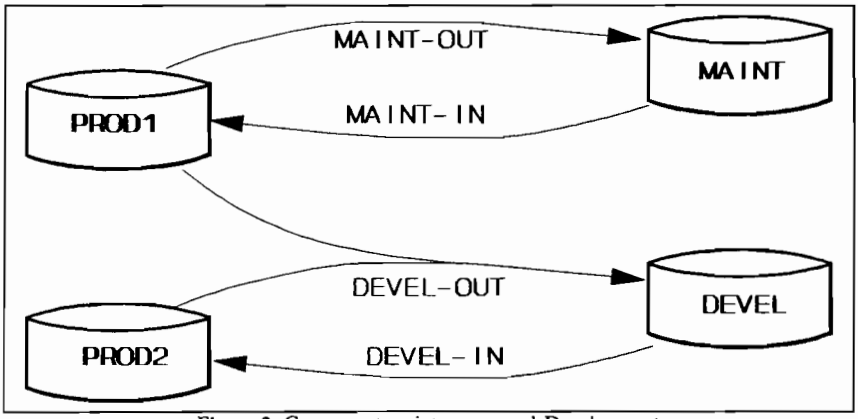


Figure 2: Concurrent maintenance and Development

Local Modifications to Package Software

Another common, but complex, situation arises when a business makes local modifications to a software package, then has to integrate those changes each time the vendor sends out a new release. Here again, the key to managing the situation lies in keeping the modified source separate from the original vendor source. The best technique for doing this is to use the MPE accounting structure by setting up a separate group for the modified source. If the vendor's software account is divided into groups by file type, it may be simpler to set up an entire new account for the modified files, with corresponding groups.

Once this structure is set up, a procedure must be established to check out files for modification from the custom group or account if they exist there, or from the vendor original source if a customized version does not exist. When the changes are complete and tested, all source files are moved to the custom "library". Object, of course, is moved to whatever location contains the executable code for the application. Most shops do not find it necessary to retain a copy of the vendor's unmodified object code.

The true test of this procedure comes when you receive a new release from the vendor, and are faced with the task of integrating your modifications into the new source. With the modified source in a separate group or account, identifying the programs that must be changed is easy. Determining what modifications must be made is not as simple. One efficient way to do this is to use a source comparison utility. Several are commercially available. The comparison utility can be used to isolate your changes to the old source. These changes can then be applied manually, or in some cases automatically, to the new source just received from the vendor. The modified code can then be compiled, tested, and moved to the modified source library.

Without a comparison utility, the best way to accomplish merging of your changes with the new source is to adhere strictly to documentation standards while making any changes to vendor code. Deleted lines should be commented out rather than being removed from the program. Changed and added lines should be clearly documented with a recognizable indicator in a particular column. If all changes are marked with a consistent indicator in the comments, a scan utility can be used to quickly list the changed lines, reducing effort and the risk of overlooking changes.

Source-object Synchronization

Source-object synchronization means ensuring that your production object code was in fact generated by compiling your production source. Source-object synchronization, combined with a good checkout-checkin procedure will prevent unpleasant surprises such as those described in "Fixing the Wrong Source" at the beginning of this article.

The most common method of ensuring source-object synchronization is by recompiling the source in a controlled location. Frequently this redundant compile is performed in the production location, after checking in the changed source. In many cases the changed object is not checked in at all; the old object is replaced in the execution location by the recompile. The major drawback of recompiling in the production library is that you run the risk of putting untested object code into production. Testing is done in the development location on object code that was compiled there; this carefully tested object is then overwritten with potentially different code by the redundant compile.

A preferred method is to perform the redundant compile in a secured test area, prior to final testing. It is important that the area be secured, so that additional changes cannot be made to source after the compile, putting it out of synch with respect to the object. With this approach, it is critical that the source and object be moved together into production. This can be done

procedurally, or by software that moves both files with a single operation, and moves neither if one cannot be moved for some reason.

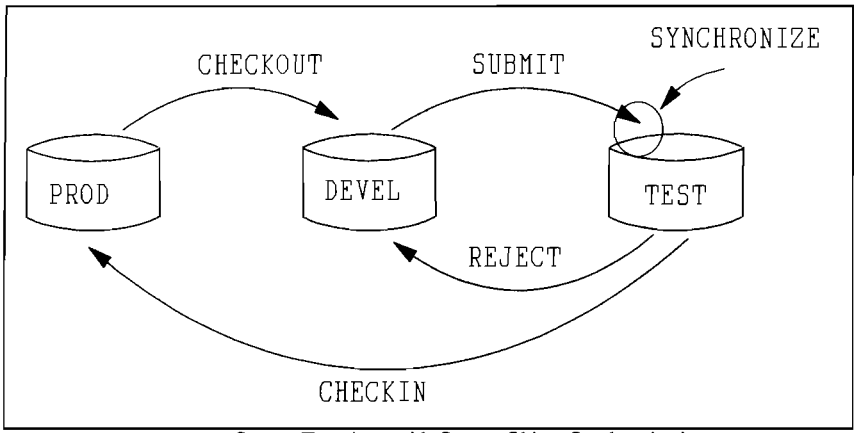


Figure 3: Secure Test Area with Source-Object Synchronization

Another approach is to mark files with a timestamp or version number to indicate synchronization. With this approach, the source and object can be compared to determine whether or not they are synchronized, rather than recompiling to force synchronization. The problem with this approach is that it is not 100% reliable unless the version number or timestamp is placed on the file by the compiling process, and stays with the file forever. A version number can be placed in the source and compiled into the object, but changing the version number is usually a manual effort, and can be forgotten. Timestamps can be synchronized by compiling, then moving the files together into the secured test area. Then timestamp synchronization can be checked again before moving the tested source and object together into production.

Another alternative is to use a MAKE utility. MAKE utilities are common in the UNIX environment, and are rapidly becoming popular on other platforms. MAKE utilities use file modification timestamps to identify object that is out of date with respect to its source - i.e., the source has a later modification timestamp than the object, indicating that the object must be rebuilt to preserve synchronization. MAKE utilities provide for definition of a hierarchy of "objects", so that you can synchronize not only with respect to source, but with respect to RL's, USL's, and source INCLUDE modules in complex applications. MAKE utilities also allow you to define rebuild rules to reconstruct the object from its components in an efficient manner, only rebuilding the intermediate components that are out of date with respect to their dependencies. Since MAKE

will identify objects that are out of synch, it is useful to run it against the production library on a regular basis. To avoid the trap of putting untested object code into production this way, the rebuild rules should be set up to compile into a Q/A location for testing rather than directly into the production object location. One final note on MAKE facilities: MAKE will only identify situations where the object timestamp is earlier than that of the source. If you move your object into production and leave the source behind (or purge it accidentally), MAKE will not detect this condition.

Managing Common Code

Version control for source programs and their related object files is relatively straightforward if you follow a structured, well-controlled change procedure. Managing versions of copylibs, source INCLUDE files, USL's, RL's, SL's and XL's, however, is much more complicated. If a copylib or INCLUDE file is changed, what source programs have to be modified, or at least recompiled? If a USL is changed, what object code must be re-built? If an SL or XL routine changes, what programs may have been changed?

The simplest approach for identifying programs affected by a change to a copylib, INCLUDE file, or callable routine, is to use a scan utility. Several are available commercially (and there is at least one in the contributed library). A scan utility is used to search through a group of files for occurrences of a particular string, which might be a copybook or INCLUDE name, or the name of a called external (SL or XL) routine. The more sophisticated scan utilities can search for several strings in one pass, and use wildcard or "metacharacter" search strings.

There are also commercial software packages which maintain a cross-reference index of dataset and data item names, copylibs, INCLUDES, called routines etc., specifically for this purpose. Alternatively, you can maintain your own simple cross-reference database, perhaps on a PC. Any of these approaches can prevent unpleasant surprises which result from neglecting to change all the programs affected by a database or common routine change.

To identify object code which must be rebuilt due to source INCLUDE, USL or RL changes, a MAKE facility as described earlier in this article is ideal. A MAKE facility can also do the segmentation for you to rebuild the object, based on your generic rules or rules specific to this program. A "low tech", but effective alternative is to maintain a rebuild jobstream for each executable file. These job streams can be scanned for the name of the RL or USL that has been changed; those that match can then be streamed.

In addition to these questions, there are "mechanics" problems, such as how to compile programs that use copylibs and INCLUDES in a test area. Should you check out the copylib and INCLUDE files along with the source? If one INCLUDE file is changed, how do you ensure that you will compile with this changed INCLUDE file but use the production version of all others?

First, many companies have found that INCLUDE files offer much greater flexibility than copylibs in managing common source code. Since each common source module is a separate file, the specific module can be checked out to make a change without tying up the entire set of common code. Moving a single INCLUDE is also much faster than moving a several-thousand-line KSAM copylib.

It is most efficient to check out only INCLUDE files that are being changed, so compiling source in a test area by pulling in INCLUDES from the production library is desirable. Read access to the production library from the development area is therefore required. Rather than having to code file equations for every INCLUDE that is not being changed, you may wish to fully qualify INCLUDE files in the source so that they point to the production library by default. That way they only have to be equated only for those that are being changed. This approach introduces some inflexibility, but a scan-and-replace utility can make short work of any mass changes that may be required.

CONCLUSION

Basic change control procedures can prevent a significant percentage of production failures, and should be part of every MIS department's operation. Change control does not have to be elaborate or cumbersome to be effective. Generally, larger shops require more sophisticated change control, but other factors are just as important. Shops with special requirements, such as maintaining local customization of supported vendor software, can benefit from more sophisticated change control techniques. Automation can improve the efficiency and reliability of change controls; you can take advantage of commercial software packages or develop your own utilities.

Task Management - A Key to Success in System Projects!

Robert R. Mattson
9545 Delphi Road S.W.
Olympia, WA 98502
(206) 736-2831
(206) 352-5038

Abstract

For most system professionals the successful management of projects is fundamental to their own success. Millions are spent each year on project management software and training. This large sum of money is spent because people know that the risk of failure in projects is large. They know this from their own experience and observations as well as from the "publicized" disasters. Why hasn't the success rate of system projects improved in proportion to the dollars spent? A fundamental reason is that we've focused on project management rather than task management! In other words, one must learn to walk before one runs. We must learn to manage tasks before we can manage projects! This talk outlines what should be done to manage a single system task such as developing a program. Masterful application of these concepts can lead to added success in system projects.

Introduction
or
"Systems Mean Projects, Projects Mean Tasks."

Software systems revolve around projects, big and small. Whether it is putting in a multi-million dollar system or making a modification to one existing program they usually involve "projects" of one sort or another. So, software professionals are involved with projects all the time.

The trouble is, our profession is having a difficult time achieving success in these projects. Research indicates that less than half of all projects are finished on time and budget. And a surprisingly large number of projects are cancelled before they are completed or even after completion because they did not deliver as expected. Even for those projects finished within time and budget, I have observed that things seldom went as smooth as one would wish.

To try to help us succeed in these myriad of projects, we look for solutions. We spend millions of dollars each year on new programming languages, new analysis tools, project management software, etc. Each of these we hope will help

us deal better with meeting our project time, quality and cost deadlines.

Yet, can we discerned great improvements in our project track record? The trade press in the last year still had many examples of missed deadlines, cancelled projects, etc. Why is this still such a problem? There are many reasons. One is that systems are getting more complex and larger. Realistically, it is impossible to know all the reasons. Nevertheless, there is clearly one area that is crucial to all the projects in which we need to improve. This is the area of task management.

Task management is the fundamental building block of project management. I've studied, applied and struggled with the techniques of project management for years. Finally, it dawned on me that almost universally this subject has been focused on too high level of view. The focus has been merely what a project's tasks are and how we manage the "project!" The focus has not been on how each of the individual tasks is defined, planned and managed. Yet, no matter the size of the project, it can and will be reduced to a series of tasks. And it is within these tasks that the project is actually done.

Therefore, this area of how we manage a single task is what I'd like to explore with you.

Why We Should Care About Task Management
or
"How Goes The Task..... So Goes The Project!"

But why should we care about task management? The key here is that you can't manage a project without managing the tasks! The length of time a project takes is equal to the time from start of the first task to the finish of the last task. The cost of a project is the sum of the costs of all the tasks done on the project. The quality of the project is the result of the quality of the work done on each task.

When a project is over budget it is because one or more of the tasks were over budget or because we did tasks for which we never planned.

One of the most common mistakes in project management flows from not understanding this fundamental relationship between project and tasks. A clear example of this is when a project manager sees the first missed task deadline and doesn't take action or assumes the project budget will still be met. My experience tells me that when this situation

occurs then the project will most likely not reach it's cost, time or quality goals. This is because the project manager doesn't understand the key relationship between project and task. Or they don't understand how to manage a task.

The conclusion is clear. If our project's tasks do not meet their time, cost and quality goals then the project cannot! Conversely, if you manage the tasks well then managing the project becomes much easier! So let's address how a software task should be "managed."

How To Manage a Software Project Task
or
"Fundamentals, blocking and tackling..." the coach said.

The old football analogy really has meaning when applied to task management. There are some task management fundamentals that one needs to understand and apply. And, as important, all the project team members have to be skilled in them as well. If these are done right then the task has a good chance of succeeding, and conversely....do them poorly and problems are bound to show-up. What are the fundamentals requirements of excellent task management? The fundamentals are:

1) Written Task Description/Deliverable/Quality Planning
2) Written Task Deadline/Time/Subtask Planning
3) Written Task Cost Planning
4) Doer's Monitoring, Replanning and Status Reports
5) Supervisory Quality, Time and Cost Review and Feedback

Let's address each of these in order.

Written Task Description/Deliverable/Quality Planning
or
"I did what I thought you wanted...give me a break."

The first place to start with any task is to describe what is to be accomplished. This is the specification of the goal, the deliverable. In systems, the challenge is to describe a task in adequate detail to insure that all important specification/quality parameters are clearly understood by both the specifier/manager and the doer.

It is easy with software tasks to NOT do even an adequate job of detailing the specification/quality parameters. This unfortunately is done too many times each day! And, if an inadequate specification is done, then all types of problems can arise. The most serious fundamental

problem is that the doer doesn't produce what is desired! In this most common scenario, the time and dollars budgeted are used up ....and suddenly one finds out that deliverable planned for doesn't exist. Then we must scramble to decided how to deal with this reality? How do we get what we need? How much time will it take and at what cost? How will we deal with our "customer"...can we hide it and "make it up somewhere else!"

Yet, identifying that inadequate task specification is a problem is one thing. The question is...how does one go about correctly specifying what is to be done? This is a whole paper in itself. In brief however, the best way I've found to do software task specifications involves defining a comprehensive set of "quality parameters." These are defined for a particular task in terms of what is unacceptable, ok and excellent. These are defined in such a manner as to clearly allow all parties involved to know what is expected in each area. Further each parameter is given a "weight" that represents the relative value if the excellent criteria is achieved. After a task is done, both the doer and reviewer will "score" the resulting product based on the quality parameters. See appendix A for a copy of the Software Quality Form that is used in this technique. For a more in depth discussion of this technique...see the paper Software Quality...Let's Discuss This Can of Worms! (Robert Mattson, Hewlett-Packard Business User Conference Proceedings, Orlando, Florida, August 1989)

A big advantage of this technique is that it is formal and written! I have found, many times, that the manager and doer can have big differences between what they each perceive the deliverable to be. But, when one has to formally write down these expectations the differences and fuzziness become much clearer. Without such a formal written approach the chances for "mistakes" and "misunderstandings" is tremendous.

A sometimes overlooked item, when dealing with a task is to describe the prerequisites that must be available for the task to begin or proceed. I've seen many tasks started; only to discover that some required prerequisite item was missing. This either made finishing the task impossible until the item was made available or required the person to take time to produce the item themselves. Planning a task should require the specification of all expected/required prerequisites.

It must logically follow only when a person knows precisely what is to be produced can they begin to determine how long it will take them to deliver the product! Yet too

often people try to do software task management without first defining what is to be done in a precise manner. Then they wonder why there is such confusion about the success of what has been produced and/or how long it should take. The lesson is that good task management starts with clear task specifications!

Written Task Deadline/Time/Subtask Planning
And Written Task Cost Planning
or
"I'll work hard on it until I'm done....I'm trying"
or
"I guess I just wasn't born a good estimator!"

Assuming we have done a good job of specifying what is to be done, our next "step" of task management is really two very inter-related steps. The first of these involves doing subtask planning, time estimation and deadline setting. The second step closely related to the first one is preparing a cost estimates. These are very inter-related because of the very close relationship between the time spent on a task and its cost. Generally, the longer something takes to do the more it costs. Somewhat contradictory is the fact that sometimes, if we spend a lot of money, we can shorten the time required to complete a task (ie. buy faster computer, new software tool, etc). The key here is that one cannot estimate the time to do some well defined task with out knowing the resources in people and dollars that one has available.

This issue of people resources mentioned above deserves more discussion. In software systems, as much as any place, there are large differences in the capabilities and productivity of people. This definitely influences the estimates of the time and cost to do any task. Differences between people are so great that without knowing who will do a task..it is literally impossible to accurately estimate how long a task will take. There is not such a thing as a generic programmer/analyst! If one doesn't know the person(s) doing the task then the only other strategy is to use one of three guess strategies...worst case, best case, and some type of average.

Let's assume we know the resources that are available to apply to the task. The next step is to figure out how long it will take, given the resources, and when we can commit to having it done (the deadline). Fundamental to this step is understanding that this estimating must be done primarily by the person assigned the task. These estimates of course are reviewed by the project manager/supervisor. Regardless

of this review, the rule is... the person who must meet the commitment makes the commitment.

Unfortunately, there is probably no harder thing for the average system person to do than estimate the time it will take for he/she to complete a task. Why is this? Partly it is due to the nature of the system tasks. The number of possible variables that can influence the time is large. It is also because very often we haven't defined the actual deliverables well enough. Still further, many people suffer from not understanding how to plan a task for which they are responsible.

I know of no magic way to solve the number of variables that can influence a task. However, a good task specification can help. We've already dealt with how to do a better job of defining the task. So let's address the fact that many system people simply don't understand how to do a task plan. This results in the following observed behaviors.

1) They don't make sure they understand what is to be delivered.
2) They don't know the prerequisites required.
3) They don't make sure they understand any imposed deadlines that the "customer" may have.
4) They don't understand the resource and dollar constraints.
5) They don't subdivide the task into sub-tasks, mini-deliverables and mini-deadlines.
6) They don't place much personal value on meeting any deadline to which they commit.

Further, the person is hampered because, in the past, he/she hasn't kept track of what sub-tasks are done on similar tasks or how long they actually took to do. Further they haven't learned that it is more important to make a longer deadline than to miss a shorter optimistic one.

Here, I'd like to emphasize the fact that too many software professionals aren't aware of what "tasks" they do or even more critically how long it takes them to do these tasks. This is compounded by the limitations in accuracy of human memory. I have personally observed how frequently people cannot accurately assess how long it took them to do something even immediately when they've finished it! Further, if a couple weeks goes by the accuracy may be off by many factors. This means someone can say it took be a couple days when it really took them a week! Look for this phenomenon yourself and I think you'll be surprised at how common it is. The ultimate cure for this will be when

system professionals log the actual tasks and times that are done for their own learning purposes.

One of the inaccuracies associated with knowing how long a task takes is what I call the "lost redo." This is the time spent after the task is "done" to actually finish it. This is usually not included in a person's memory of how long some task took to do...unless it was very large or otherwise significant. And, surprisingly, when an organization has procedures that make people keep track of their time, the lost redo is even more prevalent because it tends to be purposefully hidden! This is because, in those organizations, there are usually serious consequences to a person if they miss an estimate or deadline and so they play games with the recording function to distort what actually happened.

A discussion of this topic is not complete without dealing with deadlines. In excellent task management, a task deadline should be no longer than two weeks. If the total task deadline is longer than two weeks then either the task should be broken into smaller tasks or sub deadlines should be set. My experience shows that task deadlines greater than two weeks tend to be missed more often and have greater over-runs than those of under two weeks. These are the formal deadlines. In practical fact, excellent task management from the doer standpoint really involves "informally" setting many small deadlines associated with the sub-tasks...and meeting those.

Just a word on cost awareness. All hours estimates should be translated into dollar cost. Once we've estimated the time to do a task it is simple multiplication and addition to get the cost of the task. This will aid in giving people a good cost awareness. People who do a good job of task management seem to treat the task as if they were both customer and supplier. They tend to have good cost awareness. They have an attitude that realizes that it isn't just time that is over-run it's money.

We've now dealt with specification, sub-task planning, time and cost estimating and deadlines. Now we'll see what the doer of a task does when working on a task.

Doer's Monitoring, Replanning and Status Reports
or
The "I'm only responsible for doing the task!" syndrome

The next critical step in task management has to do with the what the doer does in self monitoring, replanning and status reporting. Notice the emphasis on the doer. To be successful with any significant project... one must have people who do task self management. The project team must all be capable of and believe in managing themselves, their tasks, and communicating to others task status. This is in contrast to the projects where only the project manager sees himself/herself as the one "responsible" for monitoring, replanning and status reporting!

To manage a task (or project) well requires frequent assessment of where the task is against the plan and specifications. This assessment is used to adjust the plan if required. Also, most importantly, the information from the monitoring and replan needs to be communicated. This communication of the updated plan allows for actions to be initiated as required to accomplish the projects goals.

How often should this monitoring, replanning and communicating be done. The guideline logic that should usually apply is as follows.

Re-evaluate, re-plan and report when:

a) 10%, 50%, 75% and 100% of the "last plan" labor hours or dollars have been expended

or

b) 10%, %50, 75% and 100% of the "last plan" elapsed time has occurred

or

c) Any factor arises which will cause quality of the deliverable to meet less than the excellent criteria.

or

d) Anything has occurred which will cause the delivery date or hours/cost to change by more than 10% of the original.

or

e) The task is completed and deliverable is delivered.

If this schedule is followed then there should be few big surprises! Tasks do not get behind suddenly at 95% complete or just when the task is suppose to be done. A motivated competent task doer should be able to see problems coming much sooner. The above schedule requires the doer to stop at key points to assess, replan and report. With this schedule you can quickly spot someone incapable or unwilling to do this. Or you can more easily see a person who is continually providing inaccurate plans and replans. Each person with such a problem needs to be handled differently. The key is that you must have people who can and will do this monitoring, replanning and communicating. Otherwise, successful projects are highly unlikely.

See Appendix B for a form you can adapt to use as the task plan/re-plan document.

I've been focusing mostly on the doer of the task and their responsibilities. This is an over emphasis for effect. The project manager must have critical involvement in all the processes described so far. Further, the next step outlined must be done by the project manager or the method will not work.

Supervisory Quality, Time and Cost Review and Feedback
or
"The manager's squeaky wheel gets attention"

An ongoing Project Manager/Supervisory review is essential to quality task management. The project manager makes or breaks the task management system. He/she makes it by reinforcing the correct process by review and providing timely feedback on the plans, status reports and software quality forms. This will keep them coming and make the process a serious part of what the doers take as their responsibility.

Additionally, the project manager is reinforced because he/she will get early warning of impending problems. This warning should give him/her enough time find a solution or at least moderate the impact on the project.

This review must be done all through the tasks existence and also most critically at the end of the task. A review should be held when the task is "done." This is where everyone involved should be able to assess the different successes and areas for improvement. On any project of significant complexity and duration, this final review process should do a lot to improve performance over the

project. Without an ongoing review, a project just seems to become hotter, harder and tighter as time goes on!

Keeping Commitments
or
"Don't say you will ....unless you will!"

All of the above techniques don't mean much if the people involved don't keep their commitments. In fact if you have people who are superb at keeping commitments ...then some of the fundamentals can be de-emphasized. My experience is that most people are less than perfect at keeping commitments. The techniques outlined helps them to focus on the importance of accomplishing every task on time, cost and quality. And it gives them the means to improve their good intentions.

In spite of everything, people will not always meet commitments. The key is how you handle this. If they perceive that it is not important to you that they missed a commitment then watch out. You've just said it's ok to not keep the next commitment. You must of course balance this with wisdom that say's no one, including yourself, always keeps their commitments. So it's a fine line a project manager must walk. Nevertheless, the bottomline attitude of your project team should be ....never make a commitment that you cannot or will not keep!

Summary Conclusions

As I re-read this paper, it strikes me that I've made the process sound like a simple by the numbers process. It is anything but this...in my experience. All management has to do with people. And no two people are the same! Somehow we must put people first while getting our projects done. At the same time management means stretching people to do things that may not come naturally. Otherwise, it's not management but more like being an bookkeeper looking only backwards.

Task Management is really nothing more than Project Management of a "sub project." Tasks are really just projects with smaller scopes, shorter time-frames, smaller costs and less resources. Project management is done by a partitioning of the project into smaller and smaller pieces and applying the fundamentals of task management to them. And, if one has project members who keep commitments and manage their tasks well then lo and behold the project will be much easier to manage.

Finally, software project tasks are complex. Managing and doing them successfully is not easy. However, with the right approach and attitudes our success in such projects can improve significantly.

The Author

Rob Mattson is currently Manager of Information Systems at WIDCO, a large open pit coal mine in the state of Washington. He has been working in the systems field for 14 plus years, the last 11 in the HP3000 environment. His background in addition to many systems related roles includes a degree in Psychology, work as a Certified Public Accountant and as e management consultant/trainer. His current professional areas of interest include general systems theory, project management and exploring the concept of system quality. When not immersed in systems he enjoys his family, sailing, sea kayaking and golf.

Paper Presented Originally at HP3000 International User's Group Meeting - San Francisco, Sept. 1989.

Appendix A

Software Quality Form

System Name: _____ Est. Time: _____
Program Name: _____ Actual Time: _____
Date: / / Assigned To: Reviewed By:

Quality Parameters	Quality Level					Rating	
	Unacceptable	OK	Value	Excellent	Value	Doer	Rev.
Functional Specifications							
Suitability Job Effectiveness							
Speed Responsiveness							
Resource Impact Overhead							
Robustness Forgivingness							
Adaptability Flexibility							
User Acceptance Satisfaction							
Business Cost Effectiveness							
User Independence Support Required							
User Documentation							
Ease of Learning							
Ease of Use User Efficiency							
Implementation Installation							
User Interface Uniformity							
Development Task Management							
Cost to Develop							
Time to Develop							
Test Plan Testing							
Technical Review Walkthrus							
Defects "Bugs"							
Maintenance Time & Cost							
Maintainability Int. Documentation							
Adherence to Standards							
Integration							

Comments: TOTALS

% of Excellent

The Goal Is Excellent Systems

TASK MANAGEMENT FORM #1												
As of:	Last Report:			By:				Status:				
Task ID:	Project ID:			Customer:				Task Resp:				
Task Description:												
Task Deliverable:												
Quality Form Reviewed & Accepted By:				Subtask Planning Done (Y or N)?								
Prerequisites:												

		Commitments			Proposed or Actual Chg From:						
		Original	~~Revised~~ Current	Proposed	Original			Revised Current			Next Update Due:
					Amt	%	$	Amt	%	$	
Labor Hours	P										@65hrs or
	A	Pre:	Cur:	Tot:							
Start Date	P										
	A										
Working Days	P										@08 days or
	A	Pre:	Cur:	Tot:							
Completion Date	P										@ 12/18
	A										
_____	P										
	A	P:	C:	T:							

Quality will be excellent in all parameters except:

Issues I need to resolve to keep these commitments:

Sign offs--> Plan: Quality: Completion: Task Mgmt Rating (1-10):

R.Mattson 1989

System Design for Dynamic Organizations

Gunnar Fredlund
Nord Software, Inc
P.O Box 60610
Santa Barbara, CA 93160

Telephone: (805) 569-5094

Introduction

Fast growing and changing companies poses new challenges for system design. Computer systems are necessary for these companies for continued growth, they are strategic tools and not primarily aimed at cost reduction.

As a consultant you often get in contact with these companies when their old systems running out of steam, seriously threatening future growth. You wish they had started the project years ago - but they did not. You wish they could give you a clear picture where the company will be next year, in five years, but they can not. Your only guarantee is constant change.

These companies often have difficulties in using standard software systems, most packages just can not keep up with all the changes. The transaction volumes can be a problem as standard software is not optimized for fast data entry or efficient storage usage. All options in the packages are necessary when adapting the software for different businesses. However, options add lots of extra disc space and makes data entry complicated.

If you decide to go with a custom made system, you are facing a real challenge. Old well-known techniques like Software Engineering does not give a solution in time. You need to prototype and start production at the same time.

For MIS these companies are difficult to handle. A failure could seriously hurt the company and success will only be rewarded by an increasing flow of enhancement requests.

> Sun Microsystem, Inc. changed from using HP3000 based order entry and production systems to a mainframe based solution. Sun lost control over production and inventory during the implementation because of software errors. The fast growing company could not sell it's newly released products. Sun's usually good quarterly

profit turned into a loss for the first time in the company history. Even worse, Sun's internal computer debacle got all attention from the press instead of the new products.

To be successful in development of systems for dynamic organizations you need to rethink the whole process from initial goal setting to final product.

Luckily enough there is ways to create software which can survive a rapid changing environment. HP's operating system MPE was developed for a computer serving only a few users, a small hard disc, punched cards and paper tape. You can still run your old programs produced a long time ago on todays big spectrum machines (but don't try to use a paper tape punch).

Design is only partly a technical discipline, you can not become a good designer by reading books. A good designer must have a certain feeling combined with technical skills. When working with dynamic organizations the feeling is very important, you will never have enough time for thorough analysis but you need to take the decisions anyhow.

One way to get going is to create a set of thinking tools. Tools you use to be able to understand the organization well enough for handling the design.

Goals and Critical Factors

The first step in the design process is to find out the goals for the system and it's critical success factors.

You need to be precise, the goal is never to get a new AR system. Perhaps the goal is to get a better cash flow, reduce labor, gain control or something else.

Different users in the organization can have different goals for the project, any conflicts must be resolved so the project has clear goals.

Critical factors are those factors which tells if the project is a success or a failure.
In an order / entry system critical factors can be

- all orders received should be taken care of in 24 hours
- the sales person should be able to see on the screen (in less than 5 seconds) if the ordered items are available
- invoices should be produced same day as the goods is shipped
- the on-line system should be available 7 AM to 6 PM all working days

...

> The Bay Area Rapid Transit system for routing and tracking BART trains uses a 20 year old Westinghouse CPU. In 1987 Logica Data Architects, Inc. started programming a replacement system with the requirement to handle 74 trains.
> Testing in November 1989 showed that 74 trains would require between 83% - 130% of available CPU capacity. The transit system has plans, to begin schedule 108 trains in 1990. The system needs to be partly rewritten to be able to run on a different CPU.
> Some BART officials wants to cancel the project, after having spent $20 million.

If a single critical factor is not met, the whole system will fail.

Evolutionary Delivery and Feedback

Traditional system development can not be used for fast growing companies. The methods gives a well defined path from project start to implementation,

> goal setting --> analysis --> overall design --> detail design --> coding --> testing ---> implementation

This software life cycle takes to long time to complete. It is also to static for all changes. At the implementation phase the company is not the same as it was during the analysis and design phase. Maintenance has to start before the system is set into production.

That is not what I learned in computer science. My text books talked a lot about models, flow charts and similar techniques. They should have recommended a supermarket idea, mark the models and flow chart with "Best before _____".

First consider the speed of the process - these companies can not wait they need the results immediately. The development process must use prototype techniques but with real data.

Start the project with an over all design. Split the final product in as many independent modules as possible. Let the organizations needs and possibilities for pay-back decide the priorities and the implementation schedule.

Do not try to deliver a complete system at once, it can lead to a disaster. There is usually no time for parallel tests or class room training. A better approach is to give the users a new module at a time, installing the system in a

step by step fashion. This approach also gives the project team time to work out software problems before the next step is released.

Fast growing companies is very dependent on the environment and need feedback all the time.

> HP is now growing faster than the industry average. However, it can not expand it's work force as a closer examination revealed that the growth is in low margin products. The sales increase resulted in freezing of new employment instead of the opposite.

The same is true for system development, constant feedback is necessary to stay on the right track. It is so much better to find out if the response time is good enough after a month of programming instead of at the end of the project.

> The Swedish car manufacturer Volvo tried to set up a CIS (Corporate Information System) several years ago. They hired a famous Californian 'Think tank' to do a feasibility study.
> They spent over 100 work-years on the project, used the most modern mainframe computers and data bases. The whole project took five calendar year to complete. Every routine worked as it should but when installing the whole system it failed to update a normal days transactions within one day. They had to give up the project as they already was using the biggest available mainframe.

If you release a module at a time and get feedback you will notice changes both in the computer system and user routines associated with the system. Often user changes affect the next modules to be coded. The system design has become an interactive process where both the company and the system changes during the process to accomplish ultimate results.

80 - 20 Rule

In a system 80% of available programs is needed but only 20% is critical for success. Normally a project team spends 80% of it's time on the non critical routines instead of spending it on the 20% critical routines.

80% of the errors occurs in 20% of all programs.

20% of the routines gives 80% of the pay-back.

Our goal is to isolate the most important issues from the rest. We can concentrate on the critical routines and then having an automated process for remaining programs.

Core System and Enhancements

Even the most dynamic organizations have a core business which stays the same while everything else is changing.

Invoicing routines can not be handled in the same fashion when you go from 10 invoices a month to 30000. From a customer view the procedure should still be the same.

When looking at history, current systems and plans you can find out what is stable and what is dynamic. Try to design a core system (often a data base and procedure libraries) based on your companies core business. Add all dynamic user routines as extensions to the core system. When the company changes you add and remove user routines but keep the core intact.

> Company XYZ started with a small AR system there all invoices and payments were manually entered. When the company later on needed an automated order / entry system sales and shipping staff took over data entry for the invoices. Further growth made the company starting using EDI (Electronic Data Interchange). Data entry is now done by the customer. The invoices are sent through data communication instead of mail. The company also decided to use their bank to process all payments and send them a magnetic tape instead of all checks.
> The AR staff thinks they have a new AR system, for VP of Finance and MIS the system is still the same.

A good illustration of a core system is a General Ledger. We are still using double-entry bookkeeping suggested by a Franciscan monk, Fra Luca Pacioli, in the fifteenth century. The core system has survived but many implementations which do not separate core from the rest are failing. Some can not handle more than one currency, others are limited to 99 divisions, etc.

You must be able to handle all enhancements with speed and without losing historic data. It requires not only a good structure for the design but also a very bright project team. The programmer who takes ownership and pride in his programs often tries to avoid changes, talking about user problems instead of business changes. You can not use that type of programmer in your team. The team needs to take ownership and pride in the whole system regardless of whom coded the actual routines.

The most frequent requests concerns reporting, new formats, selections and presentation. If you are using IMAGE on a HP3000, you can solve several request at once by adding an end user report generator to the menu.

System Design by Walking Around

You need to be familiar with the company from management to shop floor. The time between discussion and decision can be very short in dynamic companies. The company information is often spread man to man instead of using formal means.

You need to walk around and talk to people. The success of your system is as dependant on psychology as computer science. A user who feels ownership and commitment to the system can help you when a new module does not work. The same error can be an excuse not to use the system for a negative person.

With growth follows frequent staff changes. New employes have many ideas for enhancements. Some of the ideas have already been tested and reversed when the previous staff gained more experience. You need to have enough knowledge to be able to differentiate between real enhancements and learning patterns.

> In a company specialising in group travel I discovered the following cycle. Autumn was low season for flight bookings and the department discussed new control systems to assure the correctness of bookings and to keep the budget. In spring the flight department had its prime booking period. Everyone was stressed out, they did not have time to supply data for any control system and not to check the budget. After the peak season most of the staff left the department, tired of the work load. The company hired new staff. Next autumn came and the new flight department started thinking about what kind of control system they needed to stay in budget, totally unaware of what to come, ...

Simple Systems

You need to design the system so it does not require extensive training. Software system should be as easy to use as cars. When you have learned how to drive one car you can easily change to another model without reading instruction books. A user of accounts receivable should understand accounts payable just by looking at the new screens.

You have to trade features for simple straight forward screen designs making it easy to understand. Standard system typically offers lots of optional fields so you can use the

system in a wide variety of business. If some people uses them and others not it can cause a mess for routines checking the fields.

The technical design should also be as simple as possible. A customer screen should correspond to a customer maintenance program, a customer data-set, procedures for add, change, inquiry and delete. The program for vendor maintenance should be a copy of the customer program.

The most common cause for system failure is to complex systems. A human can not handle a lot of parallel activities at the same time. If you don't have an overview of the whole project, you can not see if it will work or not. It helps a lot for the understanding to split the system into independent modules. Another way to help you handle large system is to set up patterns. Use the same types of screens, same locking strategy and the same type of interfaces for all modules.

In many projects the design phase get to little time. Every one is so stressed by the tight time schedule so they try to start coding as soon as possible. My experience is that you gain a lot of time if you take the original design back to the drawing board again. Redo the design time after time until it is clean and easy to understand, making coding and testing trivial.

It is easy to draw the conclusion that the key is to have a really good 4GL tool or a relational data base. However, my experience is that the design process is the important phase. With a good design you can develop dynamic systems using COBOL and IMAGE in a short time. At the same time projects using the latest tools but without a sound design take forever to finish.

The important issue for your tools is not how fast you can create a system but how easy and fast it is to change the system.

Automatic Validation

In companies handling a stable volume of transactions you have elaborate control structures based on many years of experience. When volumes changes drastically these methods becomes obsolete. No one looks at data and discover typos in an automated information system. The validations and reporting of exceptions must be handled by the system. It is easy to check if some one is stealing from the cash register if you manage one store but with hundreds of stores you need automated routines.

The faster the company growths the more automatic control structures you need. No one has time to go through all reports anymore.

Also the control procedures needs to be flexible. One of my former clients increased sales by raising their product prices 100% and then negotiating 50% discounts. The automatic control system refused the transactions because maximum allowed discount was 15%.

The modules need good error handling so they can be implemented independently. New modules often gets data from old, not yet replaced, systems with bad data quality.

We are currently designing a control language for one customer. They will use IF , PRINT and WARN statements to test and report any data problems. The system extracts information directly from the data base without using the application software. We chose a control language instead of using a report generator to simulate a manual audit of the data.

```
IF STATIONARY(SALES,CURRENT-MONTH) <
     STATIONARY(COST-OF-SALES,CURRENT-MONTH) THEN
          WARN("Check cost of sales!")
IF BINDING(SALES,CURRENT-MONTH)  < 0 THEN
          WARN("No binding reported!")
```

The audit statements will be placed in a batch file and processed after every month end. The system will read and audit the entire data base once a month.
Our goal is to end reporting used to find errors, the tedious manual audit process and at the same time improve the quality of the data.

Summary

You need to design the system to allow

- immediate implementation of selected routines
- step by step development
- easy handling of structural changes
- implementation without extensive training
- immediate follow up
- automatic control procedures

In the design process you have little help of existing methods as they applies to stable environments. You have to use common sense, experience and thinking tools.

A Bottom-up View of Top-down CASE

Abstract:

Many implementations of CASE for business systems developers are based on the principles of Information Engineering (IE). IE uses the metaphor of a pyramid to depict a top down approach starting with strategic planning through logical analysis to physical design to system construction. While many system developers appreciate the potential benefits of computer aided design and construction, they are less certain of the usefulness of "Upper CASE" concepts and tools. This presentation will address the ways the "Lower CASE" tool user can use strategic planning and business analysis tools to contribute to the overall quality of the application systems which support a business enterprise.

Prepared by:

C. David Key
Ernst & Young
2175 North California Blvd.
Walnut Creek, CA 94595

(415) 977-3903

Introduction

Engineering is a discipline. "Information Engineering" (IE) is a specific discipline for determining the information management needs of a business enterprise and developing automated systems which satisfy them. In the mid-1980's our firm made a policy decision to adopt the principles of Information Engineering, to build a system development methodology based on them, and to acquire or develop the software tools required for the practical application of the science of IE. The policy decision has paid off, both for the firm and for its clients. The feedback from practical applications of IE has led not only to stronger tools and higher productivity but also to a greater appreciation of the real benefits of the approach -- namely, the greater likelihood that the systems developed will fully meet the needs of the business.

Unfortunately, the discipline has not always been welcomed by a group who might benefit most from it -- professional systems developers. In fact, this phenomenon -- resistance to productivity -- was one of the stimuli for choosing this topic. Another stimulus was our recent success in using "reverse engineering" technology on existing business application systems.

CASE

CASE is a widely used acronym. Unfortunately, it doesn't always mean the same thing to everyone. Some would have that any computer aided system development tool -- a compiler, an editor, a debugger is a CASE tool. Others hold that "CASE" stands for Computer Aided *Software* Engineering -- that is, it can be applied to the development of even a single program. While I accept that proposition, the "CASE" I'm talking about today is Computer Aided *Systems* Engineering. In other words, the engineering of entire business application systems.

The Repository

For CASE to work there must be a common repository of systems information which covers the entire development life cycle.

Just as with application development tools, this concept has been steadily evolving. As long ago as the 1950's, System Development Corporation developed the "COMPOOL". This was a set of standard data names which was maintained by a small group in Santa Monica. If you were one of the three thousand programmers working on the SAGE Air Defense System, you had to call headquarters and "order" a data name. Many years were to pass before this idea of a "Data Dictionary" became commonly accepted. Those of you who use one appreciate the major impact on productivity and system quality which it offers.

With CASE, the Data Dictionary has evolved to become the System Dictionary and then the Encyclopedia and finally, the Repository. The Repository (whatever it is called) is the home for all the information which defines the information needs of a business enterprise, the design of the systems which support those needs. *Using today's tools and techniques, every line of code required by an application can be generated from the Repository.*

Information Engineering

Traditionally, Information Engineering is represented by a pyramid depicting four phases. Strategic Information Systems Planning is at the top and is concerned with determining the overall information needs of the enterprise. Business Area Analysis is next and is focused on a specific business area. The analyst seeks to develop logical models of the data available to the business and the processes necessary to satisfy those information needs which were identified in the planning phase. Application Design translates the logical models into physical representations of screens, reports, data bases, files, programs and other computer specific system components. Finally, moving to the base of the pyramid, the Construction

phase converts the design to a fully tested business application system.

Another way of looking at the pyramid is that Planning is concerned with the "why" of an information system, Analysis is concerned with "what" it should do, Design deals with "how" to do it and Construction gets it done. Regardless of one's viewpoint, it is important to understand that, in IE, each of these four phases has a precise definition with clearly specified objectives, tasks and deliverables.

Looking up from below

To many application developers, the bottom of the pyramid is grounded in reality and the higher one climbs the more esoteric it all becomes -- leaving Planning (at least as defined by IE) somewhere in the lofty mists of theory. Most will move up the pyramid a little way and concede the benefits of at least some Design -- laying out screens and data base structures before one begins programming is probably worthwhile. Fewer will include program structure charts and action diagrams on their list of necessary system development ingredients, preferring to sketch out a general design and then "wing" the programming. (The politically adept may refer to this technique as "prototyping" thus making it sound high-tech and productive. In fact, prototyping is a formal part of the IE Design phase and should not be confused with mere trial and error coding.)

Moving further up the pyramid to the Analysis phase, we find even fewer who understand and use logical data and process models. Logical models are, by definition, independent of a specific physical computer environment. This is the characteristic which make them powerful. It also seems to be the reason that they are often not used: "We don't have time to do logical modeling, we have to get the system installed!" A reasonable, if rhetorical retort is, "If you don't have time to do it right, how will you

find time to do it over?" If there were an accurate logical model for each application system currently in production, getting out of the current morass of old systems, wedded to old technology, would be much simpler.

If IE Analysis is hard to sell to system developers, imagine what happens when you get to the top of the pyramid. Planning, like all IE phases, is an activity with formal rules. It is a process whereby information is gathered about the goals, strategies and resources of a business enterprise. These facts are assembled in the repository where the relationships among them are recorded. Other facts and opinions ranging from the organization chart and the functions performed by the various units to the problems which might be encountered and the factors which will be critical to business success are added to the repository. Using a combination of automated and manual processes, these facts are collated, correlated, weighed, massaged and analyzed. The deliverables from the Planning phase may include such documents as a Technology Strategy Document, an Information Needs Assessment or a set of system development priorities.

Upper & Lower CASE

Partly as a play on words, partly as a reflection of relative positions on the pyramid, and partly as an allusion to the levels of an organization to which they seem to appeal, Planning & Analysis are often referred to as "Upper CASE" phases with Design & Construction known as the "Lower CASE" activities. However subtle, this too seems to have had an effect on the opinion which Lower CASE practitioners hold regarding the Upper CASE disciplines.

"I already know what I have to do and why!"

The arguments for not using Upper CASE tools and methods are fascinating if only for their entertainment value and predictability. I hear developers (notice I seldom mention programmers) say, "*I don't need a data model, my data base is already designed.*" This is often from the same individual who is having to add a data structure (element, segment, or view) to the "already defined" data base.

"*The SISP has nothing to do with me, I just do what I'm told.*" The days of the programmer who doesn't understand the business but just produces code are fast disappearing. Asking what business need a program satisfies is a legitimate question from a developer. It is very much an Upper CASE question. (Those who don't ask can expect the question to eventually be raised by their successor).

"*This job is too small for all that nonsense.*" Given the power of today's utilities, ad hoc query facilities, and user-friendly software, it is safe to say that there is no development task requiring programmer or analyst time which is not part of the enterprise information model.

"*I'll spend more time planning and analyzing than I will coding.*" All the statistics suggest that the majority of the time spent in application repair is due to failure in requirements analysis and specification. In other words, if the Upper CASE tasks are done well, the Lower CASE tasks are a relative breeze.

"*It sounds great but I don't really understand data flow and entity relationship diagramming.*" This is a legitimate problem. In order to make use of Information Engineering and CASE, you must have tools, you must have a methodology and you must have training. If your company is unwilling to make this investment, you may find yourself doomed to using 1960's technology into the 1990's.

Build the repository and then use it!

Years ago, in a *Datamation* article on programmer productivity, Daniel McCracken said that the two keys to achieving productivity were:

1) "Do a given thing only once."

2) "Keep everything in the same place."

He was talking about a library of reusable subroutines and, possibly, a data dictionary. The same principles, however, apply to CASE. If you are using a full life cycle CASE tool to design and develop your programs, then you have a repository available for Upper CASE information.

Deliberate vs. casual planning

While "casual planning" may sound like an oxymoron, CASE tools make it possible. When you are designing that next application, allocate a day or two to identifying the Business Goals, Data Entities, Critical Success Factors, Organizational Units, Business Functions, and other formal Planning concepts which are implied by the existence of such an application. Use the CASE tool to define the associations among these objects and include the resultant analyses as part of your design documentation.

Do the same thing for the next application. Or, if you want to really make a mark, do it for an application that's already in production. What you will have begun is the development of an Enterprise Model for your business. You will soon discover that you and management are starting to ask the right questions before starting that next project.

Painless Analysis

As long as you've been willing to devote a little time to capturing Planning objects, go a little farther. Step back from your application a few paces. Use the CASE tool to draw the Data Flow Diagrams which put it in a business context. Describe the data flows, processes, data stores and external

agents using Entity Diagrams. Use Decomposition Diagrams and Action Diagrams to further specify the processes (This should include the non-automated as well as computerized processes.) You should find that these exercises help you to understand and communicate the services your application provides. At the very least, it's easy to do and the diagrams look nice. In many cases, you'll find a pitfall or an opportunity that might have been overlooked. Most importantly, you will be building the repository.

Redevelopment Engineering

Over the past year or two, we and others have been having a fair amount of success in "reverse engineering" -- using computers to derive program specifications from code, develop data models from data base definitions, produce system documentation from production libraries, and similar "Construction to Design" processes. One of the most promising avenues for reverse engineering is the automatic population of a CASE repository. This offers a significant head start to those of you who may be intrigued by casual planning and painless analysis.

The "Golden Spike" Project

Moving down the Information Engineering pyramid from Planning to Construction has become known as "forward" engineering to distinguish it from reverse engineering (above). A major California aerospace company is planning a project which encompasses both. Two independent groups will address one of the organization's mission-critical applications. One group will interview users and executives (Planning) and use the latest Information Engineering Methodology, to "forward engineer" the application requirements. The other group will start with the currently installed version of the application and reverse engineer it to develop the requirements which it appears to satisfy. Then they will compare notes. (The project's name derives from the story of

building America's transcontinental railroad.)

We expect to find that the users have information and service needs the installed system doesn't meet. As interestingly, we also expect to find that the current application offers capabilities of which the users (and probably MIS management) are unaware.

Summary

I mentioned earlier that CASE and Information Engineering require three ingredients: tools, methodology and training. The tools must utilize a common repository which spans the business application development life cycle.

There is a fourth ingredient which is ultimately necessary -- management commitment. This presentation however, is taking a "bottom up" look at CASE. (If it were the conventional top down view, I would be urging you as managers to use the most powerful productivity tool I know -- an arbitrary decision, and get started!) My suggestion, therefore, is that you:

1. Convince management to make a modest investment in tools and training;
2. Use the Lower CASE tools to design and generate your applications;
3. While you're at it, sneak some Planning and Analysis objects into the repository;
4. Use the diagrams and CASE analyses as part of your system documentation;
5. As you begin to gain new insights into the functions and services your systems provide, discuss them with your boss;
6. See what happens next.

INFORMATION MANAGEMENT TECHNOLOGIES INTO THE 1990'S

Orland J. Larson
Hewlett-Packard Company
19091 Pruneridge Avenue
Cupertino, California 95014
(408)447-1197

ABSTRACT

Enterprises have come to depend more and more on the accessibility, accuracy and timliness of information. During the next few years, new database technologies will provide solutions to many of the problems and difficulties facing today's MIS and user departments.

This paper begins by reviewing the changing world of information management and the challenges facing MIS. This will be followed by the major trends associated with information management over the next several years. These include the continued acceptance and importance of relational database technology, the increased interest in distributed database applications and the emergence of the new object-oriented database management capabilities. Finally, this paper will address the significance of these technologies in the cooperative computing environment.

INTRODUCTION

THE CHANGING ROLE OF DATA PROCESSING

Before looking forward into the 1990's, we should first look back at how data processing has evolved over the past twenty years.

The general trend of the 1970's was the use of centralized computers and resulted in systems that were often difficult to use, inflexible and usually did not meet the end user's needs. Database management systems (mostly hierarchical and network) became widely used and provided the basis for on-line, interactive applications. In addition, the computers and operating systems provided programmers the capability of developing applications on-line, while sitting at a terminal and interactively developing, compiling and testing these applications. The end users were also provided easy-to-use, on-line inquiry facilities to allow them to access and report on data residing in their databases.

During the 1980's, the emphasis has been on the decentralization of data processing. This includes the proliferation of personal computers which has resulted in both the "islands of automation" and the corresponding "islands of information" problems. This in turn resulted in reduced control of corporate data for the MIS department. In addition, relational databases became commercially viable and experienced wide acceptance even though performance was often an issue. Relational database performance has now improved significantly; and they are currently proving effective in on-line transaction processing (OLTP) environments. Software tools such as 4th generation languages (4GL's) continue to be used successfully as an effective way of developing applications through the concept of information systems prototyping. This required that the end user be more involved in the development of systems and has resulted in more effective systems that

meet the users' needs. This has helped to reduced the backlog of applications but usually also has contributed to the "islands of automation" problem.

As we move into the 1990's, relational database will continue to gain wider acceptance. It is the enabling technology and the basis for distributed database management, which provides transparent access to data which is distributed over several sites.

There are also new technologies called object oriented database (OODB) and object oriented programming systems (OOPS), which will manage more complex data structures and will result in improved programmer productivity and more flexible systems.

An additional technology, cooperative processing, is evolving which will help integrate those "islands of automation" back together and allow for the data and programs to be accessed and shared in a cooperative computing environment.

THE IMPORTANCE OF SQL

According to a recent Gartner Group Report, in 1988 only about 7% of the applications developed used relational database or Structured Query Language (SQL). However by 1992 their prediction is that 65% of applications developed will use SQL.

There is no doubt that SQL will be the basis for applications developed in the 1990's. One of the main advantages of SQL is data independence or the immunity of applications to changes in storage structure and access strategy. Another main advantage of SQL is the simplicity of the underlying relational model which is the easiest to understand - at least at the most basic level. In this model, data are represented as tables, with each horizontal row representing a logical record and each vertical column representing one of the attributes, or fields, of the record.

The following are the key points associated with relational technology:

* Relational concepts are easy to understand and use.
* SQL is a multifunctional language
  - Database definition and creation
  - Data retrieval and manipulation
  - Authorization and security
  - Transaction management and recovery
  - Database environment management and restructuring
  - Interactive and programmatic use
* SQL allows you to specify which information you want - not how to retrieve it.
* SQL increases programmer productivity and raises programming close to the level of problem solving.
* Data independence is ensured and minimizes maintenance of programs
* Data access can be automatically optimized as the DB structure changes.
* The DBA has power and control over the database.
* New systems can be implemented much faster.
* SQL assists in cross-system connectivity.
* Relational databases provide a cost effective, powerful solution.
* Basis for a true distributed database environment.

There are, however, some areas of SQL that need improvement. The current SQL standard is weak and missing many important features and many of the standard features are implementor defined. In other words, no vendor fully supports the complete "standard" and no two SQL implementations are exactly alike. The SQL language has many inconsistencies and the same SELECT command may yield different results on systems from two different vendors.

As we enter the 1990's, relational database has become a dominant technology in today's information management marketplace. There are several enhancements planned to improve functionality and performance. It eventually will be appropriate for most applications and gain wide acceptance by all users.

Relational databases can improve the quality, control and accessibility to your organization's extremely important and valuable information resources. It can result in an improved competitive position by aiding business analysis that can help to determine ways to improve products and services.

Unlike non-relational database environments, relational databases adapt easily to dynamic business requirements. In addition, unrestricted access to important data means better information for more effective decision making.

Relational database can also have a positive effect on many MIS development environments by reducing the application backlog and reducing the time and cost required to develop applications. The improved database flexibility and ease of change can also result in a significant reduction in the maintenance of applications.

Overall, the use of relational technology can increase the MIS professional's effectiveness and productivity, which results in improved user satisfaction and confidence. Choosing relational now will position your organization to take full advantage of the technological advances of the 1990'S.

DISTRIBUTED DATABASE

One of the hottest topics in the commercial database world is the growing trend towards the use of distributed database management systems. After many years of research, distributed databases are becoming more viable. However, there is still much to be done to provide more than just read access to distributed data. Chris Date, one of the world's leading experts on relational database, recently presented a paper (see reference 3) in which he provided a working definition of distributed database. "A distributed database system is a system involving multiple sites connected together in a communication network, in which each site is a database system in its own right, plus a user at any site can access any data in the network exactly as if the data were all stored at the user's own site. Thus, a DDB is a virtual DB whose components are physically stored in a number of distinct real databases at a number of distinct sites."

Chris Date follows this working definition with an "alternate" or more elaborate definition. "A distributed database system is a system that allows an arbitrary collection of relations, from an arbitrary collection of databases, on a variety of different machines, running a

variety of different operating systems, connected by a variety of different communication networks to function as if they were all stored in a single database on a single machine. The user is completely insulated from all details of distribution."

Distributed databases can allow the structure of the database to mirror the structure of the company, while simultaneously solving the "islands of information" problem. Some additional advantages include local control of local data, accessibility to remote data, increased capacity, incremental growth, data availability, efficiency of storage, flexibility and cost effectiveness.

There are also some potential problems or disadvantages associated with distributed database such as the complexity of implementation - but this is the vendor's problem. Some additional potential problems include the problem of how to design systems for distributed environments, the complexity of administration and control, the impact on local operations, the political problems dealing with the ownership and protection of the data and the possibility of a node or line "crash". In addition, solutions that are appropriate in a centralized environment may frequently not be appropriate with distributed systems.

Chris Date is the author of 12 rules of a distributed database system. He begins with the fundamental principle or "rule zero" that states, "To the user, a distributed database system should look exactly like a nondistributed system". The subsidiary rules follow:

1. Local autonomy
2. No reliance on a central site
3. Continuous operation
4. Location independence
5. Fragmentation independence
6. Replication independence
7. Distributed query processing
8. Distributed transaction management
9. Hardware independence
10. Operating system independence
11. Network independence
12. DBMS independence

These rules are fairly self explanatory and will not be expanded in this paper. No current DBMS vendor adheres to all of these rules. A few vendors claim adherence to rules 1-8 and almost none to rules 9-12.

David Wilde, Project Manager at Hewlett-Packard's Database Lab, recently spoke at the San Francisco INTEREX Conference and presented HP's phased approach for developing a distributed database management system. The following is a summary of that approach:

* REMOTE DATABASE ACCESS (ALLBASE/NET) - Program can read/update a remote DB without coding for communication and remote processes.

* FOREIGN DATABASE ACCESS (e.g. IBM's DB2) - Program can access multiple vendors databases without coding for DBMS differences.

* MULTI-REMOTE DATABASE ACCESS - Program can read and update more than one remote database at a time.

* SNAPSHOTS - Enables a user to copy all or part of a table from one

database to another, optionally this table could be refreshed.

* PARALLEL QUERY EXECUTION - Ability to execute multiple queries to multible databases at the same time.

* DISTRIBUTED UPDATES WITH TRANSACTION MANAGEMENT - Enhanced transaction management to support updates to multiple DB environments.

* REPLICATED DATA - Enhanced availability and performance through multiple copies of data, with automatic synchronization of copies.

* PARTITIONING - Enhanced performance and availability through partitioned tables.

Some of the issues regarding distributed database that will have to be addressed in the 1990's include:

* Distributed query optimization and decomposition
* Fragmentation, recombination and optimizability of data
* 2-phase commit and recovery
* Referential integrity across sites
* Management of replicated/partitioned data
* Controlling authorized user access
* Update synchronization
* Degree of Transparency
* Flexibility to move data around the network
* Cost of mainframe vs. mini vs. micro MIPS
* Provision of foreign (non-HP) DBMS gateways (gateways are a way of processing data in a foreign DBMS or file system)

There are some factors that will help propel the distribution of data. These include company mergers, the downsizing of computers, the increased database needs and the general industry push toward distributed and cooperative processing. A company with distributed operations will gain competitive advantage through the support of distributed databases.

OBJECT-ORIENTED SYSTEMS

Each decade, one or two key advances emerge to change the practice of software development. Object-oriented systems and methods are rapidly entering the mainstream of software engineering and systems development. Leading consultants are heralding object-oriented approaches as one of the most important trends to affect businesses in the 1990's. But even among its strongest advocates, disputes abound over key issues, content, and definitions of the object-oriented approach. Object-oriented technologies are moving out of the academic world and into the business world.

With the object-oriented approach, processes revolve around the data, not the other way around. Using the traditional approach, programs are structured around data rather than procedures. For example, when using a traditional programming language, parameters are used to pass data structures and values between routines. The object-oriented approach attaches routines to data structures. In other words, the behavior of the data is kept with the data. This is called encapsulation.

An object-oriented programming language allows the programmer to define and manipulate objects. Some object-oriented programming languages are

extensions of classical languages - C++ and ObjectiveC are in this category. Others are brand new languages, eg. Smalltalk and Eiffel.

An object-oriented DBMS also supports the definition and manipulation of objects, plus providing the classic DBMS functions of persistent storage, transaction management, concurrency control, security, backup and recovery.

A "message" is an important concept used with the object-oriented approach. It is defined as a request sent to an object to change its state, or to return a result to the sender. Objects respond only to well-defined messages. The only information needed to use an object is knowledge of the messages it can receive. An object-oriented program is a flow of messages among cooperating objects.

Messages ensure the modularity of a system. To interact with any object, you only need to know what messages to respond to, not how the object is represented. Knowledge of how an operation is accomplished is of interest only to the programmer responsible for the definition of the object itself. Messages make an object's functionality available to other objects, while hiding the implementation details.

Maintaining and modifying software has been a real drain on programming resources. Maintenance programmers must understand a complex system well enough to fix its problems or enhance it. However, these programmers often did not develop the code, and often operate without adequate documentation or guidance. Changes often introduce new, unanticipated problems to the system. A programmer working with existing code must read and understand it; this may require a mental translation back into the original design specifications, which is extremely difficult for complex or poorly-coded systems. An additional concern is that programming languages allow unchecked access to data structure internals.

Objects can dramatically improve the problem of maintenance. Modularity and encapsulation limit interdependence, allowing changes that do not disrupt the rest of the system. Objects' natural organization make it easier to learn and understand relationships between parts of a system. The original programs are easier to write and debug and fewer errors occur. Programs read like designs, making changes clear and easy to make. Reuseability is an important advantage of using objects and libraries of these software components provide leverage. What has been written once need not be written again. Model features like inheritance allow existing components to be incrementally modified to suit changing needs. Together the representational advantage of encapsulation and the features of inheritance dramatically improve software development and can greatly improve programmer productivity.

Libraries of high-quality, tested software components will radically alter the way software is written. Software will routinely consist of a series of software components glued together. Application programming will no longer mean rushing to a text editor to begin coding. It will require understanding the capabilities and restrictions of available components, plus knowing how to combine them.

Some of the challenges facing object systems include the time it takes to learn about existing libraries of software components. Programmers also may resist accepting this new approach. Objects also consume more resources, however emerging 80386 PC's and RISC workstations will help

to alleviate this problem. Applications with promising potential include: prototyping, user interfaces, graphics, telecommunications, computer aided design (CAD) and computer aided manufacturing (CAM).

Object oriented products are still in their infancy and the commercialization of object-oriented technology has barely begun. Now more suitable for advanced technology projects, object database systems should become viable for commercial projects over the next few years and widespread adoption in the 1990's.

Hewlett-Packard has developed a prototype of an object-oriented database management system (OODBMS) called Iris. Development started in 1985 and Iris has been presented and demonstrated at several major conferences in the past few years including: SIGMOD (Special Interest Group on the Management of Data) in June 1988 in Chicago, OOPSLA (Object-oriented Programming, Systems, Languages and Applications) in September 1988 in San Diego, DB/EXPO in February 1989 in San Francisco and at the Patricia Seybold Forum on Object-oriented Technology in April 1989 in Boston.

Object-oriented environments herald the dawn of new programming paradigms. Business people will be empowered to perform tasks that, in the past, required professional programmers. Programmers will be empowered to design complex applications in smaller, modular, more fool-proof pieces.

Neither end users nor application programmers will need to concern themselves with the mechanics of networking, peripheral support, or file handling. Object based architectures lend themselves to the creation of a much richer information environment. Digitized voice, music, images, video clips, and animation will begin to populate our information systems.

COOPERATIVE COMPUTING ENVIRONMENT

The environment in which today's business must operate is changing quickly and becoming more complex. To meet the challenges produced by this changing environment, organizations need greater amounts of information to make the key decisions required for success. Keeping pace with the rapid change that is occurring means gathering information and making decisions faster than ever before.

As we look at the computer industry, we are about to embark upon the next revolution in computing. This is not a unique case. There have been multiple revolutions in the computing industry and computation in general, going back to the early days of mechanization, tabulation and so on, through the first computer mainframes. The mainframe was a very centralized processor, still oriented toward batch and was really a carry-over from punchcard tabulation systems. The next move was into the distributed processing and personal computer revolution, causing an explosion of workstations and personal computers on people's desks, which fueled distributed processing and distributed computing.

Then came the communications revolution, where the objective was to integrate all this computing power in the corporation in a way that moves information around and enables different types of devices to

participate in the solution of business problems. This has become the basis or launching point for the next revolution of cooperative computing.

Cooperative computing is the notion of tying together all the information computation resources in a corporation into a single entity, and making all those things interact in some efficient manner, transparent to the end-user in such a way that each user has access to all the information computation resources in the network as though they were local to the user's workstation.

The cooperative computing environment is Hewlett-Packard's vision of the future of computing: a network of heterogeneous computers that can work together to solve a single problem and are extremely easy to use.

To support the cooperative computing environment requires a cooperative computing strategy. HP has focused its product development efforts in three key areas - systems, easy to use workstations and multivendor networking. And perhaps most importantly, the integration of the three.

The corresponding three technological pillars of HP's strategy are:

* HP Precision Architecture - HP's powerful, flexible architecture that serves as a common foundation for HP computers. It includes an advanced, optimized and integrated application development environment that allows programmers to be more productive.

* NewWave Environment - Simple, natural consistent user interface, which can provide one "window" into the entire network of computing resources.

* HP AdvanceNet - HP's networking strategy based on standards, both industry and defacto.

This paper has addressed many of the important technologies that are essential to the cooperative computing environment. ALLBASE/SQL is the strategic DBMS of the future for cooperative computing. ALLBASE/SQL runs under both MPE and HP-UX. The application development environment for both of these platforms is an excellent 4th generation language called ALLBASE/4GL and an easy to use end-user oriented inquiry and reporting facility called ALLBASE/QUERY. ALLBASE/SQL will also be the basis for distributed database technology in the future. ALLBASE/NET is the first phase of HP's distributed database technology and uses HP's networking capabilities.

The concept of "objects" in the NewWave environment is similiar to the object-oriented capabilities described in this paper. The object management facility (OMF) is the main component of the New Wave Environment. The OMF tracks all data in the PC, whether it be text, graphics, spreadsheets, scanned images, even voice. These objects are represented as icons which can be combined into compound documents containing different types of data.

The OMF allows users to create "hot-links". Hot-links allow users to share data between different reports, memos, even file folders. When data is changed in one place it is automatically changed every other place it has been shared.

The OMF knows what application to use to work with a particular object. When an object is selected, the NewWave environment will automatically load the proper application.

SUMMARY

The important technologies briefly addressed in this paper: SQL, distributed database management systems and object-oriented systems are extremely important to the future of the cooperative computing environment. HP's Cooperative Computing Strategy is focused upon helping our customers meet the challenges of today's changing business environment. We believe that these technologies and strategy, coupled with HP's commitment to industry standards and reputation for high quality, reliable systems, can provide our customers with the solutions they will need into the next generations.

REFERENCES

Connors, T. and Lyngbaek, P.,"Providing Uniform Access to Heterogeneous Information Bases",Software Technology Laboratory,Report STL-87-16, Hewlett-Packard Laboratories,December 4, 1987.

Date, C.J.,"Distributed Database Technology", Proceedings DB/EXPO'89, San Francisco,CA,January 31-February 2,1989.

Eunice, J.,"Leaders in Object Databases",D.H. Brown Associates, Inc., December 1988,400 Benedict Avenue,Tarrytown,NY,10591,(914)631-7859.

Fishman,D.H.,Beech,D.,Cate,H.P.,Chow,E.C.,Connors,T.,Davis,J.W., Derrett,N.,Hoch,C.J.,Kent,W.,Lyngbaek,P.,Mahbod,B.,Neimat,M.A., Ryan,T.A.,Shan,M.C.,"Iris: An Object-Oriented Database Management System",ACM Transactions on Office Information Systems,5(1),Jan 87.

Larson, O.J.,"Strategic Importance of Relational Database Technology", Proceedings INTEREX Computing Management Symposium'89,Nashville,TN, March 8-10, 1989.

Tash, J.,"Application Architecture and the Strategic Role of Relational DBMS",Database Decisions, Inc.,41 Marcellus Drive, Newton,MA,02159, (617)332-3101.

Schussel,G,"Database Developments in the 5th Generation",Proceedings 1989 Database and Cooperative Processing Symposium,Washington,DC, March 20-22, 1989.

Stein,J., and Maier,D.,"Concepts in Object-oriented Data Management", Database Programming and Design,1(4), April 1988, pp.58-67.

White, C.,"The Importance of SQL",Proceedings DB/EXPO'89,San Francisco CA,January 31-February 2, 1989.

Wilde,D.,"Distributed Data Management From HP",Proceedings INTEREX HP Users Conference,San Francisco, CA,September 11-14, 1989.

IMPLEMENTATION MANAGEMENT

Applying Project Management Techniques to the Implementation of Application Software Packages

Jane Fastiggi
Hewlett-Packard Company
8080 Pointe Parkway West
Phoenix, Arizona 85044
(602) 273-8132

Much has been written about project management as it relates to the management of software development projects. But today, many - if not most - HP 3000 customers purchase rather than write their major application software. The implementation of this application software must be managed just as carefully but it frequently is not. This paper deals with the use of formal and informal project management techniques in the implementation of application software packages. What follows are recommendations on how such a project should be managed, and the final section contains examples of what commonly goes wrong and why.

Why Project Management

Project management is a set of formal techniques for the planning, organizing, directing and controlling of a project. Project management has been used for many years by large aerospace and defense companies to manage multi-year and multi-million dollar development projects. It has become increasing popular in recent years in businesses of all types and sizes, and especially for data processing projects. This is because projects have become more complex and - more importantly - because the projects that organizations attempt today cross the traditional hierarchical management lines. It is this situation - the implementation of an integrated manufacturing and financials package or a payroll and personnel application, for example - that makes project management critical to success.

The Phases of a Project

The formalization of project management has two main aspects: a definition of distinct phases, tasks and milestones involved in the project, and a set of techniques for the management and control of the project. Each aspect is equally important.

Formal project management in a data processing development environment involves the following phases:

* Project Definition, in which the project and its scope are formally defined;
* Feasibility Study, in which it is determined whether or not the project can and should be done;
* Systems Analysis, in which the overall project is outlined and planned;
* Design, in which the detail design of the project is done;
* Programming, in which the actual coding and unit and systems testing are done;
* Test, in which the user-acceptance tesing is done;
* Installation, in which the end-user training and data conversion are done, and the system goes into production; and
* Maintenance and enhancement, in which the system is used and changed as the needs of the users change.

Upper management is involved heavily in the first three phases. The end-users are involved heavily in the design, test, installation and maintenance phases, but upper management stays involved throughout in a supporting role.

When an application package is purchased, the same phases exist, but in a different form; these phases are as follows:

* Project Definition is the same - the project and its scope must be carefully defined;
* Feasibility Study becomes the search for an application package that meets the company's needs for an affordable price;
* Systems Analysis is the phase in which the implementation process is defined and planned;
* Design, programming and test are eliminated - they are the phases that take the most time, and this is the main reason that people purchase rather than write their own software;
* Installation includes not only training and conversion but also such things as rewriting the chart of accounts and developing procedures;
* Maintenance and enhancement involves the integration of new features and use of report-writers.

Here again, upper management must be involved throughout, and the end-users are involved in the later phases.

Much attention is paid to this aspect of project management. Support tools for this aspect, like Gantt and PERT charts, are widely available. All too often, project teams focus on the milestones and tasks but forget that the techniques associated with project management are as critical to the success of the project.

Project Management Techniques

The techniques that are used in the traditional data processing project are important. Much attention is paid to how a software development project is managed because the costs are so enormous; too often this is ignored when software is purchased. But the "turn-key" application is just a mythical as the "man month"!

Many techniques are used throughout the software development and installation cycle. Those that need to be applied to the implementation of purchased software are as follows:

* Each phase must be clearly defined, with milestones and tasks identified and responsibilities for each task assigned.
* The phases are not linear but cyclical; that is, new facts discovered in one phase often require that the team go back to the previous phase or phases and make changes. For example, during the feasibility study it is often determined that the project must be redefined; it cannot be accomplished with the time, people and money available, for example.
* The project scope must be carefully defined and priorities set by highest level management at the outset of the project.
* The project team must be identified early on, and the core team must remain intact throughout the implementation, including the maintenance and enhancement phase. This includes assigning a project manager and giving him or her the responsibility and authority to manage the project.
* Upper management must be committed throughout - and must communicate this commitment to all levels of the organization.
* A detailed plan must be in place before any work is done (in the case of purchased software this means before the software is installed). This plan is a working document; it is monitored and enhanced on a regular basis.

* Communication is critical, and this is best accomplished by regular project status meetings.
* Resources must be identified in the analysis phase, and must be kept accountable throughout the project.

Most companies use these techniques in the software selection process. A clearly defined goal is set, software packages are carefully evaluated and the best one selected from a vendor who can and will provide the proper levels of support, training is scheduled and taken, and then the system and the software is installed. Then why do things seem to go so smoothly for some organizations and so painfully for others? Because that is just the beginning. The follow-through techniques that too many people ignore must be applied. The remainder of this paper focuses on the techniques that are the most valuable - and the techniques that are most commonly ignored.

The Project Team and Project Manager

Selecting the project manager and project team is the first step. The project team must be in place during the first phases of project definition and software selection and remain intact throughout. The project manager must begin managing the project from inception through installation.

The Project Team

The project team consists of a high-level manager, representatives from all departments involved, and the project manager. The high-level manager must be one to whom the department heads of all departments involved report, because his or her main role is goal setting and conflict resolution. This manager must be committed to the project and communicate that commitment to everyone in the company. In other words, this manager is the project sponsor.

Two members from each of the involved departments are part of the project team - a management team member who will be involved heavily in the software selection and a lead user who will be involved heavily in the actual installation. The management team member makes sure that the goals and needs of his or her department are met, and is responsible for allocating the needed resources from his or her group. The lead user is responsible for conversion to the new system, end-user training, the development of procedures, and the maintenance and enhancement phases.

The Project Manager

Care must be given to the selection of a project manager, and his or her responsibilities and authority must be clearly defined and communicated to everyone involved. The main responsibilities of the project manager are:

* development of the implementation plan;
* monitoring and controlling the implementation;
* team status meetings;
* setting priorities;
* organizing all resources.

In other words, the project manager doesn't really DO anything. For example, the project manager does not attend training, unless the budget allows - the lead users attend the training.

Some do's and don'ts in selecting a project manager are:

* DO select a strong person with good leadership skills; the project manager leads the team.
* DO select someone with good verbal and written communication skills; the project manager leads meetings, documents and communicates progress, and communicates with vendors, which is more effectively done in writing.
* DON'T select a user from one department if multiple departments are involved, especially if the departments do not now cooperate and work together closely and smoothly (many payroll and personnel departments, for example).
* DON'T use a computer programmer; they are detail-oriented, and it will look like MIS "owns" the project.
* If you must go outside the company, DO look for an industry consultant in your industry.
* DON'T use your hardware vendor, unless the person assigned is an industry consultant; hardware vendors generally don't know anything about your business or your software.
* DON'T use your software vendor, unless the person assigned is an industry consultant; software vendors want to get the software implemented, and you want to get the application implemented.

Planning and Controlling the Project

The key to a successful project is a good plan, and the key to a good plan is to start early and constantly enhance the plan. Start developing the plan before the software is selected. Asking for implementation information during the

software evaluation phase not only helps get a good and complete plan in place early on, but also can help in the evaluation of vendors. Don't think that planning is not necessary because you are replacing an existing automated system with a new one; this is more difficult than automating a manual system!

Project Planning

A project plan starts with a strong and complete statement of goals. Next, major milestones are identified, each with a start date and an end date. Each milestone is then broken down into tasks, the tasks into subtasks, and the subtasks into actual work packages. The smaller units also have start and end dates; in addition, the interdependencies are defined and the work packages are assigned to individuals. Remember that the project plan is a working document; when a work package is not completed on time, the entire plan must be reworked. But keep the plan in perspective; no plan can ever be set in concrete, and if completing the plan on-time and as initially defined becomes too important, the result will be the famous "we did it on time and on budget - but it's totally useless". Schedules are made to be broken, and project plans constantly evolve.

Controlling the Project

Projects are controlled and directed through project meetings. The bad news is that project meetings must be held, and held on a regular basis with all members in attendance. The good news is that project meetings do not have to be painful. Project meetings should be held once a week and should last no more than one hour. The only purpose of the project meeting is to review the status of the project - period. No issues are resolved in the project meeting, but they are identified in the meeting. Projects are like sharks - if they don't keep moving, they die. Meetings keep projects moving.

The project manager is totally responsible for the meetings. He or she prepares an agenda, and distributes it at least one day in advance along with the updated project plan. A good meeting format is for each department manager to give a brief status report, along with any issues that are impeding the department's progress. The project manager takes notes, moves the meeting along, and develops an action plan. The meeting is over at this point. The project manager distributes the notes and action plan, and works with the appropriate team members outside the meeting to resolve the issues before the next meeting. This makes for short,

effective meetings and also reinforces the roles and responsibilities of the team members - the project manager is responsible for the plan and the team members are responsible for the work.

Identifying and Using Resources

In most projects, all of the necessary resources are available, but are neither identified nor effectively used. The project manager is responsible for coordinating the resources, both internal and external. The responsibilities of all resources, both internal and external, including the hardware and software vendor, must be clearly identified, monitored and controlled.

Internal Resources

Internal resources are usually clearly identified, and the main issue here is ensuring their availability. This is one of the main reasons that management is part of the team. The project manager must have the authority to ensure that the team members meet their commitments and must be able to resolve any conflicts that arise. The project manager's responsibility is to "run interference" for the team members. For example, if a user from the payroll team needs some information from a user in the personnel department and is having trouble getting it, the project manager is responsible for resolving this. The project manager has the authority to go to the user or the user's manager to get the communication problem fixed.

External Resources

External resources are usually not as clearly defined, especially at the outset of the project. The external resources that should be available are as follows: the hardware vendor, the software vendor, training courses, implementation services, and telephone assistance. The sooner all available resources are identified, the early they can be used. Here again, the project manager is responsible for ensuring that commitments made by and for external resources are met, but first the project manager must understand what resources are available.

The project manager is also responsible for running interference between the team and the vendors. Because the vendors do not work for the company directly, the most effective way of dealing with them is in writing. Identify problems and document meetings and commitments in writing

immediately after the problem is identified, the meeting is held or the commitment is made. This can insure that all parties have a common understanding of the situation and the problems are attended to in a timely manner. And yes, your software and hardware vendors can actually meet with the project manager and the team at the same time!

External resources are often ignored or used at the wrong time by the wrong people. These external resources include training, implementation services, and telephone assistance.

Training

The right people must be trained at the right time. The lead users should attend formal classroom training at the beginning of the installation phase. It is often helpful for the project manager to attend training with the users, but it is not necessary. Formal classroom training is recommended because it gets the users out of the work environment and provides an opportunity to learn from the other students.

After the training is completed, the users return to the site and use what they have learned to fine-tune the project plan and to develop procedures for how the software will be used. In the formal training they learn how the software works; in developing the procedures they determine how the software will be used to accomplish the tasks of their departments. These procedures are documented and form the basis for the on-site training of the other users by the lead users.

Telephone Assistance

Telephone assistance is very valuable, but it is no substitute for training. It should not be used until formal training of the lead users has been completed. Each lead user should be the telephone contact point with the vendors. Each call should be handled by the lead user responsible for that area of the application, and should be carefully documented by that user.

Telephone assistance should be used not just to report problems but also to answer questions on software use. Don't wait until you are hopelessly confused and under time constraints to place the call, but also don't expect the telephone consultant to solve all of your problems. You will receive an overwhelming amount of documentation with your system; one of the best services that telephone

consulting can provide is to point out where in the many manuals the information that you need is located.

Implementation Services

The implementation services provided by your hardware and software vendors should be clearly defined during the software evaluation process. At the very minimum, the vendors should provide assistance in the development of the initial project plan, including the setting of major milestones with realistic start and end dates. Some vendors provide project implementation handbooks and assign one of their support people to your project team. These are good things to look for if the implementation is your first.

Take advantage of these services even if you have an MIS group or have implemented software packages before. Also look to the community of users of your software package; other users are normally very willing to share their experiences with you. Take advantage of every source of experience available - at the very least it will confirm that you are on the right track and have the project well under control.

Project Management in a Dynamic Environment

Most businesses are constantly changing, and often the implementation of a new software package accelerates this change. Therefore, it is extremely important that the project team be firm yet flexible. Changes come in two forms - changes in the business impact the software implementation, and the software implementation often changes the business itself.

When Your Business Changes

Often, changes in the business require that the project be modified to take new directions into account. The project team must have an understanding of the business and recognize when this is required. It often means reworking the project plan and adding to the project team. In very dynamic businesses, this should be recognized at the outset of the project, and the appropriate checkpoints for re-evaluation of the goals and scope of the project should be part of the project plan.

The project should not be put on hold unless absolutely necessary, because it will be extremely difficult to pick it up again. If this does happen, the project team must go

back to the beginning once the project becomes active again. This means redefining the goals, reworking the project plan, and often changing the project team itself. Here again, a strong statement of goals and active involvement of the project sponsor are crucial. No one should be penalized because the business climate or the business itself changed, and this will be easier if the directive comes from the highest level of management.

When Your Software Changes Your Business

Often the business changes because of the software implementation. Many businesses use the implementation of a new application package to make needed changes to the way that business is done. Many companies have changed outmoded accounting practices or changed the focus of a business (from manufacturing-driven to customer-service-driven, for example) through the introduction of new software. This is often the easiest way to change the business, but it is very difficult on the project team. In this case, a very strong project manager is a must. An outside consultant is a good choice for project manager in this case - everyone can hate and blame the project manager, who will leave after the system is installed.

Changes are a given in today's business world, but the project team must take changes into account at every phase of the project plan.

What Works and What Doesn't

The vast majority of projects succeed - some are just much more successful than others and at a much lower cost. The most common mistakes made delay the implementation and make it very painful for the end-users; these mistakes are detailed below.

* No clear statement of goals.
  This results in a project for which the success cannot be measured; no one is really satisfied, because expectations were never clearly set. It also results in a project that goes off in many different directions, with each group setting and attempting to achieve its own goals. The single most important success factor is to have highest-level management set the goals and communicate these goals to the team and the entire company.

* No project manager or the wrong project manager.
  The project manager must have strong leadership skills, and must have the responsibility and the authority and the resources to control and direct the project. Often the project manager is either too high in the organization (not enough time) or too low in the organization (not enough authority). The lack of a project manager or the wrong project manager results in a project without direction or control.

* The wrong project team.
  The project team should consist of the project sponsor, the project manager, and at least two representatives from each department involved in the implementation. The manager of the department must be on the team, because only he or she can allocate resources and resolve conflicts. A lead user for each area of the application must also be on the team, and must be an individual who understands the day to day job functions and can train and lead the end-user implementation.

* No plan, a too high-level plan, or a plan without defined phases and start and end dates.
  This results in an implementation that drags on forever - half the people don't know what they are supposed to do, and the other half let what they are supposed to do slide because of the more pressing concerns of the day to day tasks. It also results in an unrealistic plan; the milestones can never be achieved in the time alloted.

* Training and the development of end-user procedures ignored.
  This happens most commonly in environments with an MIS department; the users lean heavily on MIS, and MIS is used to training and procedures coming naturally from the design phase. This leads to frustration on the users' part, with the software never really being utilized effectively - the users sort of figure out someway to get their work done, often by fighting and "kluging" their way through the application. This also happens when the wrong person - typically the project manager - is trained instead of the lead users.

If implemented correctly, an application software package can be what it is advertised to be - a quick and easy way to make people more productive. If implemented incorrectly or haphazardly, an application package can cost just as much and take just as long to implement as an in-house developed application.

Use the project management techniques: define your goals, assemble a good team, assign the right project manager, develop a good plan, follow the plan, use all of your resources, and make sure at each phase that the project is meeting the current company goals. But keep in mind that you are working with people, and that most projects make significant changes in how people do their jobs. The success of a project is not that it comes in on time and under budget but that the time and money spent was worthwhile for the company.

References

Project Management for Small and Medium Size Businesses
Harold Kerzner and Hans J. Thamhain
Van Nostrand Reinhold Company, publisher

Project Management Handbook
David I. Cleland and William R. King, editors
Van Nostrand Reinhold Company, publisher

Winning at Project Management
Robert D. Gilbreath
John Wiley & Sons, publisher

APPLYING EXPERT SYSTEMS - PART II

by Karen M. Hopmans

Brant Technologies Inc.
2605 Skymark Avenue
Mississauga, Ontario
L4W 4L5
(416) 238-9790

At last year's Computing Management Symposium in Nashville, I gave a presentation called "Applying Expert Systems in the Commercial Environment". At that presentation I discussed several expert systems currently implemented by large companies and pointed out the resulting cost savings and other benefits these companies enjoy.

For example, the American Express Authorizer's Assistant summarizes sixteen terminal screens of cardholder information into a single screen to help a human authorizer perform his or her job more efficiently and accurately. Authorizer's Assistant has cut the time of a typical transaction by 20% to 30% and has contributed to a 75% reduction in bad judgements.

Ford, General Motors, McDonnell Douglas and others have expert systems that perform diagnostics, helping the user of the system to quickly locate and/or repair problems.

And of course, there is everyone's favourite: Cooker or "Aldo on a Disk", Campbell Soup's successful attempt to retain human expertise as a corporate asset. Aldo Cimino retired several years ago as Campbell's resident expert on the hydrostatic and rotary cookers that kill bacteria. Instead of allowing a storehouse of knowledge to take years of valuable experience away when he retired, Campbell and Texas Instruments worked with Aldo to create an expert system to diagnose and solve problems in the same way he did.

These applications are all very interesting, but many people have said to me: "That's great - for a company that can afford to spend millions of dollars on R & D. But what about my company/my division? My total annual budget is only $_____. Does that mean expert systems are not for me?

Absolutely not. There are many small businesses using expert systems successfully and reaping benefits similar to those of the corporate giants - but with minimal start-up investment.

The purpose of this paper is to demonstrate not only how expert systems are being implemented by smaller companies, but also how to begin using expert system technology in such a way that it:

a) meets your budget and
b) becomes a key part of your long term corporate strategy.

Expert systems are a subset of the broader field known as Artificial Intelligence (AI). Simply put, expert systems are computer programs that model the reasoning of a specialist in a narrow domain of expertise. In other words, they represent in software form the knowledge of human experts.

The following expert system applications have been implemented (or are under development) by a range of small to medium sized companies - or small divisions within larger companies:

Expert Sales Advisor for Distribution Management

- front end to Order Entry system
- aids sales and telemarketing personnel during conversations with prospective clients
- supplies the user with up-to-date information on pricing, promotions, inventory and products (based on nature of conversation with client)
- benefits of the system include sales cost reduction and employee training

Personnel Scheduling System to Estimate Future Overtime

- will monitor current and future workloads for about 150 people and produce an estimation of future overtime needs
- will also match employee preference for overtime working hours and produce a schedule

Sales Support Expert System for Mixing Design

- selects and positions submersible mixers for different fluid handling situations (configuration)
- the user converses in dialogue fashion with system to determine process specification, fluid properties, installation data, etc.

- result is presented in a report recommending mixer combination(s), installation, positioning, etc.

- other uses for this application include paper pulp mixing and biological treatment systems

Title Insurance Expert System

- further automation of the process of searching public records to ensure that a parcel of land is free from legal encumbrances

- prioritization of information in a manner emulating human title searchers who are "experts" in the field

- goal is to make personnel up to three times more productive and to ensure greater management control

Job Skills Expert

- an employment agency receiving numerous resumes every day required a system to match candidates' job skills with available positions (as closely as possible)

- results in faster placement and less manual labour in searching files

Poultry Expert

- works with an existing poultry environmental control system to diagnose diseases in the broiler house

- the diagnosis is based on environmental conditions such as temperature, relative humidity, ammonia levels and feed and water consumption

- as the birds grow, the system alerts the grower to potential diseases, based on past environmental conditions

Computerized Inspection Monitor

- diagnoses and fixes problems in a computerized inspection system consisting of eleven separate components related to a single processor

- many components can fail, making it difficult for unskilled technicians to diagnose a problem

- the expert system will allow any technician to easily troubleshoot the entire system

The above examples illustrate how expert systems can be applied in many diverse application areas. It is an interesting and exciting technology that broadens the types of problems we can solve in our computing environments. But how do you start up an expert system project without blowing your budget?

I believe that the methodology is clear and logical. Once you have decided that you are the expert system "champion" within your company, read everything you can get your hands on. I have listed in the appendix to this paper some of our favourite sources, but there are many more. Your local university bookstore is an excellent place to start your research. And don't limit yourself to textbooks. There are many industry and business publications that have recently published articles on expert systems. Most libraries will provide

indexes to back issues of major periodicals.

Now you are armed with a fair bit of knowledge about expert systems and understand terms such as "shell", "knowledge engineering", "heuristic" and "object oriented programming". The next step is to begin a broad-based, low-cost training program for all levels of your company's staff, from managers to front line workers and sales personnel. This will ensure a common conceptual understanding of expert system technology and perhaps even the beginning of a corporate commitment for the future.

The training program may consist of video-based courses or "live" presentations by experienced expert system consulting firms as well as by companies using expert systems today - most people love to talk about their successes!. Other possibilities include training courses in specific expert system shells or languages (although these may be somewhat expensive), expert system building workshops, and more.

The greater the internal resistance to the technology, the more valuable it is to gear the training program to the particular "hot buttons" or knowledge bottlenecks currently existing within the company. Find courses that use as examples applications that are relevant to your own environment.

Once the initial education phase is completed and there is general acceptance of expert system technology, take a subset of your most important potential application and build a prototype. This may be done internally by staff with previous experience, or you may wish to enlist the aid of a consulting firm. The latter option will enable your people to learn as they watch and participate in the development of the prototype. This pilot project should take six to eight weeks to complete and should cost $20,000 to $30,000, depending on the nature of the

application and the time and resources available.

It may be a good idea to have an outside consulting firm undertake a preliminary feasibility study, for both the prototype and the eventual system. This should consist of a day or two of discussions and will result in a documented evaluation of the potential and scale of the overall project, before you spend the initial $20,000. This is yet another way to ensure (and insure) the eventual success of your expert system initiatives.

So, now you have a prototype (preferably on a PC) that you can demonstrate and test by fielding it for a set period, perhaps running parallel to the "old" system. During this evaluation time, keep track of the cost savings in man hours as well as savings resulting from increased consistency. These are the two key benefits of commercial expert systems.

Before embarking on an application, even at the prototype stage, you must ensure that the project will likely make a good expert system. Following is a list of features that characterize an expert system and may help you to qualify whether or not a candidate application will lend itself to an effective solution by this technology:

- √ the problem area is characterized by the use of expert knowledge, judgement and experience

- √ human expertise is not or will not be available on a reliable or continuing basis

- √ the task is decomposable, allowing relatively rapid prototyping of a small subset of the complete task

- √ the task is teachable to novices (therefore teachable to and by a computer)

- √ there are recognized experts solving the problem today

- √ there is an expert willing and available to work on the project

To increase your chance of success, you should consider these points before introducing an application as a candidate for expert system technology.

You should not try to build expert systems to solve problems people are unable to solve. Nor should you try to solve problems that are effectively solved using other techniques.

Our philosophy at Brant is "don't think of expert system technology as a solution looking for a problem. Rather it is an alternative way to solve a problem domain not effectively addressed by traditional programming tools.

Following the methodology outlined above, by the end of your first year you should achieve at least the following:

1. General education and acceptance of expert system technology and its benefits.

2. Experience in building a prototype system using expert system development tools, knowledge engineering techniques, etc.

3. A proof of concept working prototype to be used and monitored for several months.

4. A corporate commitment to expert systems as a key to strategic growth (and hopefully a larger budget).

Expert systems, as we have seen, can be successfully applied to commercial applications with substantial payoffs and benefits. Smaller companies as well as large corporations are taking advantage of this technology, and I should point out, the reality behind many of the systems in place within larger companies is that the expert system program began as an idea inside a small department. The idea caught on and was "sold up" internally, through prototyping and cost justification. Unfortunately it is the large, expensive projects that receive the most press.

Barbara Sanders, Director of AI Systems and Planning at General Motors confirms this by noting: "The larger-scale applications with the big payoff will come from the people who developed their confidence by building smaller systems."1

The tools and expertise exist today to help you begin applying expert systems in your commercial environment. If you start small and follow some simple guidelines, you can successfully implement this technology. This will increase your understanding of how expert systems can positively impact your business and give you the experience to work toward larger projects.

1 "Now, Live Experts on a Floppy Disk" (Fortune Magazine, October 12, 1987).

APPENDIX

Recommended Reading

W.R. Clockson & C.S. Mellish, Programming in Prolog (New York: Springer-Verlag Berlin Heidelberg, 1984).

Feigenbaum, McCorduck & Nii, The Rise of the Expert Company (New York: Times Books, 1988).

Hayes-Roth, Waterman & Lenat, Building Expert Systems (Menlo Park: Addison-Wesley Publishing Company, 1983).

F. David Peat, Artificial Intelligence: How Machines Think (New York: Bean Publishing Enterprises, 1988).

Donald A. Waterman, A Guide to Expert Systems (Menlo Park: Addison-Wesley Publishing Company, 1985).

EXPAND YOUR MIS RESPONSIBILITIES NOW

PAPER NO. 3029

A D BENNETTO
MANAGER INFORMATION TECHNOLOGY SERVICES
THE LAW BOOK COMPANY
44-50 WATERLOO ROAD
NORTH RYDE
SYDNEY,AUSTRALIA
02 - 887 - 0177

3RD ANNUAL INTEREX
COMPUTING MANAGEMENT SYMPOSIUM
MARCH 12-14 1990
CAESARS PALACE
LAS VEGAS,NEVADA,USA

ABSTRACT

MIS managers must now make the move to seek extra responsibilities in their roles, as departmental and distributed processing trends evolve, or face erosion of seniority. In the years ahead, MIS managers must endeavor to expand their skills and be encouraged to participate in different roles within their companies.

Initially, the paper sets forth the reasons for developing a career in addition to MIS, followed by a discussion of the nature and importance of correct career selection. Next, through the author's experiences of managing both MIS and Customer Service departments, a case study of the relationship between these two is presented exploring similarities, applicability of skills and benefits to the company.

This paper provides insight into the future role of MIS and the necessary steps that need to be taken urgently to position the MIS manager more favorably in the organization and to transfer their unique skills to other areas.

PURPOSE

The purpose of this paper is to question the future role of the MIS executive and to illustrate through a case study where extra functional responsibilities have added corporate wide benefits to the MIS function.

1.0 Background

1.1 Why should MIS professionals expand their responsibilities ?

1.1.1 Implications of the decentralization debate on MIS

The rapid evolvement of personal computers and the development of sophisticated software tools enabling end users greater manipulation over mainframe stored data has continued to further the decentralization of corporate computing.

A mixed strategy of centralizing certain core computing activities, in particular operations and 3rd generation programming coupled with 4GL tools for end users and the flexibility of printing exception reports locally, theoretically, can work well.

As a result of this trend, the traditional DP department will have less influence over some development projects owing, in part, to the already established tools that allow end users flexibility in retrieving small volumes of data in well defined processes for decision making purposes.

Consequently, innovative users shall be given more recognition by senior management whilst giving way to a tighter MIS department, characterized by any activity that requires cost effective, central control mechanisms. Attention may well be focused upon a range of duties including administration of computer hardware, planning for obsolescence and providing guidelines on security as opposed to direct project development.

Even though the MIS Manager and his/her department will continue to be a vital key to successful business practices, their responsibilities will demand a different set of skills.

The trend is appearing in many organizations already, where once a traditional DP department of say, twenty to thirty programmers, is being replaced by a central information core of non-skilled EDP people apt at communicating and guiding the end user population. An example illustrating this trend is the decision of one of Australia's smaller and newest entry financial institutions to implement a company wide personal computer network utilizing all "off the shelf" software. Some 500 PC's are managed by only a very small team of in-house professionals.

Many DP managers who have spent most of their adult lives in the technical womb of computers shall become increasingly frustrated and will feel threatened with these changes, as they

perceive their roles of lesser consequence.

1.1.2 Desire for greater challenges for MIS executives

Have you ever seen the diagram depicting an enthusiastic employee climbing the proverbial corporate ladder, only to reach the top rung and discover that there is no way else to go but down ?

MIS executives need challenges, stimulation, somewhere to continue their career growth or face stagnation. This equally applies to all Information Systems staff who are generally highly ambitious, career-minded individuals who tend to get bored very quickly.

To illustrate why senior computing professionals may wish to seek extra challenges, one has to examine the computer growth stages of the company and its relationship with motivation of the IS Executive.

There are four distinct categories of a company's stage of computer maturity. These are :

a). Early success through troubleshooting and generally raising the profile of information technology.

b). Proliferation of use by all and sundry to allow greater access to information.

c). Control of this proliferation through management guidelines, policies and centralized support.

d). Mature use of technology by sophisticated users.

This cycle is ever present and at any one time, many projects or people are moving in and out of one or more of these stages.

Consequently, when sufficient people and projects have reached stage four through the overall leadership and direction of MIS executives, the triumph of further similar challenges may become somewhat slow and tedious. For example, once the implementation of an online distribution and manufacturing system is installed, the thought of maintaining and improving this system may not be greeted with thunderous motivation.

The MIS executive should exercise all available skills to procure greater job satisfaction and be cognizant that some of these skills could equally be well employed outside direct MIS involvement.

1.1.3 Exploit MIS Generalist skills

Most MIS managers have well developed "generalist" skills applicable to most management positions. MIS managers are seen by others within the company, including top management, as a generalist with multi function job experience who knows the business. Lets expand on these generalist skills and what exactly they are :

a). Leadership

Leaders inspire! Actively participating in co-ordinating project teams, leading discussion on strategic technological issues, managing a department are all relevant activities for the ongoing motivation of staff and colleagues.

b). Innovativeness

The MIS executive combs the organization and its competitive environment for opportunities and initiates improvement projects through clear decisive, lateral thinking.

c). Communication Skills

MIS managers are constantly communicating with a breadth of different people either in formal discussions, presenting proposals to senior management or explaining problems to users. He/she transmits information to outsiders on organization plans, policies, actions, results etc and is able to negotiate readily.

d). Wide angle view of the company

The MIS executive, through his/her positioning in the company, is in the ideal chair for presiding over the various parts of the company and how they inter-relate.

Furthermore, the MIS executive is capable of conducting long range planning and taking into account a variety of input relating to trends of technology, corporate objectives and financial targets. A melding of these factors is necessary in providing the wider vision.

The MIS executive can employ these generalist qualities into other areas of the company as a :

a). Business Executive

The IS professional can channel his/her entrepreneurial flair into several key targets for the company. With focus on corporate objectives, work projects undertaken are constantly aligned to achieve proposed aims.

b). Planner

Some MIS managers are members of executive or long-range planning committees. As partners in planning, instead of objects of planning, MIS managers benefit from and contribute to longer term decision making.

c). Financial Adviser

Through the large investment of hardware and other resources, they provide advice on payback on projects and recommend long term costing analyses.

d). Lover of change

Promote, understand, listen to people who are experiencing first hand change. This technique is used to challenge new ideas, spread motivation and to lead groups to more prudent and acceptable decisions.

1.1.4 Summary

It is evident that winds of change are now blowing through MIS departments. Effects of decentralization and end user knowledge of personal computers will continue to change the shape of MIS.

These executives can expect to continue to expand their career paths by accepting the challenge and utilizing their generalist skills to greater advantage in other parts of the company.

1.2 How do MIS professionals expand their responsibilities and into what areas?

Firstly, the MIS executive must determine which area best suits his/her level of skills. Mostly, areas such as Distribution, Customer Service, Administration, Logistics or Manufacturing seem attractive as they consist of a complex array of systems and procedures which entail technology.

There is no handbook on the best way to expand the MIS portfolio, apart from suggesting initiating career guidance discussions at performance appraisal interviews. This will endeavor to alert your superior of your quest for extra challenges, outside the mainstream of MIS.

2.0 Case Study - MIS and Customer Service at the Law Book Company

2.1 Introduction

The Law Book Company has been established since 1898 and is the foremost publisher of legal information in the Australian marketplace. The company employs about 250 people and has regional sale offices throughout the country.

The Law Book Company is part of the world wide organization of Thomson Corporation, one of the international leaders of professional publishing with sales turnover of around 6 billion dollars.

Our customer base is predominately solicitors and barristers with approximately half of our sales coming from one state, New South Wales.

2.1.1 Organizational Structure

There are three directors of the company and five functional managers that comprise the senior executive group. This committee contains representatives from Finance, Publishing, MIS/Service, Sales, Marketing, Operations and Manufacturing.

MIS reports to the Finance Director as per traditional hierarchical structures. Late in 1988, the customer service function became responsible to MIS, rather than directly reporting to the Finance Director.

2.1.2 Brief History on MIS and Customer Service

Our MIS, like most other organizations has been dogged over a number of years through inadequate planning and lack of communication to senior management. As a result of this, only recently has the company discovered that information is an important resource for competitive advantage. MIS runs third party software covering all aspects of distribution, order processing, financial and sales analysis.

Customer Service, on the other hand has always been a lost cousin, deposited somewhere furthest out of sight as possible without any thought that service could generate new business and improve the health of our existing client base. This department performs all order processing, in-bound telephone queries and other administrative duties.

2.2 Relationship between Customer Service and MIS

"The successful MIS professional will be serving customers both directly and indirectly. The others will be looking for a different career path" (quote by Dan Roberts, VP of Ouellete & Associates)

This statement truly reflects the nature of the relationship MIS and customer service should have in common - i.e serving the customer. MIS customers are obviously internal, however the philosophy should remain the same.

Even though the nature of the respective employees in both departments are significantly different, a strong empathy is ever present. At the core of all of this is the customer and our responsiveness to his/her request.

Customer Service has always been a intense user of the system as they exploit most aspects of our software, encompassing order entry, accounts receivable, sales analysis, purchase order and inventory information. All of this information is vital in responding to real time requests of customers or speed paper work processing. As a consequence, a great deal of support and training is needed on a regular basis from MIS.

Customer service rely heavily on MIS support for uptime, technical assistance and system analysis skills to enhance procedural bottlenecks.

Contingent of this senior management merger has been the awareness and commitment of the interface between the two departments.

2.3 Similarities between MIS and Customer Service

The principles and philosophy behind both departments are strikingly similar.

2.3.1 Listen carefully

System Analysts must listen carefully to users to clearly define the proposed project or problem. The same equally applies to customer service representatives responding to telephone requests from customers. The skill of asking "open ended" questions to seek information pertaining to a request requires skillful navigation.

2.3.2 Develop good relations

Being courteous is essential in developing a good rapport with users and developing conducive working relationships within a team environment. By simply smiling on the phone, customer service representatives and MIS can instill greater commitment towards total satisfaction of the customer.

2.3.3 Respond quickly

One of MIS's objectives is to respond quickly to all user requests and to keep users informed of any change of status on bugs, enhancements or proposed changes. Similarly, Customer Service must keep their customers informed of progress of orders which have either been back ordered or misplaced (hopefully not!). At all times, the customer or user must be kept fully abreast of the status of his/her request.

2.3.4 Work group organization

Both departments are split into distinct categories for ease of processing work. Occasionally, work teams are created with specific briefs either to enhance a particular system or fix a problem. Both work groups are sometimes intermingled, depending on the nature of the request. Skills required in decision making in teams are similar across these two departments.

2.3.5 High Pressured

Being the front line, Customer Service is constantly under the "hammer". It is a real time situation which needs real time answers. MIS in some instances is not that precarious in nature, however they are continually trying to achieve project deadlines. Additionally, with staff shortages of the industry both in EDP and Customer Service in Australia, more burden and stress is placed on existing staff.

2.3.6 Both service oriented

Both departments have the service mentality. We are either serving the internal customer or the external one.

2.3.7 Both backlog situations

Invariably, backlog of work occurs through a variety of reasons which can be quite alarming.

2.3.8 Measuring goals

MIS professionals produce and manage hard data. MIS sometimes undermine productivity by not demanding measurable standards from their staff. Both departments had suffered from vague goal setting and subsequent tangible measuring of performance at a middle management level.

2.4 Applicability of skills

We found that a wide range of skills that applied to MIS fitted accordingly with Customer service. Some of these were :

2.4.1 Systems Analysis/Problem Solving Techniques

Owing to complex home grown developed systems, we were able to quickly identify and isolate bottleneck situations and propose suitable solutions. Some of the work concentrated on streamlining the order to keying processing cycle through removing a lot of unnecessary work practices.

A close examination of several other key retardants to speedier response were also significantly overhauled.

2.4.2 Capture important information and display it

We were able to capture certain vital statistics regarding the status of backlog work by person, customer activity and order to dispatch turnaround times. This information was collated regularly and graphically displayed back to staff.

By measuring only what is important and not capturing ineffectual information (which was being onerous to the employee in recording), we were able to obtain a better understanding of our business.

2.4.3 Team Building

Through motivation, opening up the communication channels, regular briefing sessions and recognition to success, we were able to build a strong team.

2.4.4 Knowledge of technology

Through our knowledge of the computer system and technology generally, we were able to devise new systems incorporating automatic tracking of orders turnaround, computerized filing systems and the optimization of other office automation systems.

2.5 Company Benefits

2.5.1 Competitive Edge

Our competitors have not attempted this co-existence and relationship between MIS and Customer Service. We believe that by focusing on technology and service in this manner, provides closer understanding of the customer needs imparting clear competitive advantage.

2.5.2 Retention of MIS Executive

By allocating extra responsibilities to the MIS executive, the company was assured that the executive would be retained for an extended period.

As MIS staff is very difficult to recruit, this provided two real benefits. Firstly, the executive was re-motivated to pursue other key objectives and secondly, no momentum was lost on MIS projects due to any premature departure.

2.5.3 Injection of energy and focus into an area in which it was needed

Customer service was an area that needed new energy and focus that would deliver quality, cost-effective service to our customers. By applying the skills of a MIS professional to this area, we were able to streamline and shorten many processing cycles in a very short period.

With more focus on systems and technology, the delivery of system improvements led to cuts in overall order lead times .

2.5.4 Higher profile of Customer Service

As a result of our efforts in this area, we were able to strengthen communication lines between many key areas such as Sales and Marketing and indicate the importance of procedures, policies and a clearly defined set of guidelines could have on the performance level of customer service.

Customer service was shown as an example of departmental co-operation, removing any previous image of the lost cousin syndrome, as it was now attached to the new breed of technology and the forefront of new system development.

2.5.5 MIS more closely aligned to Customer Service

The major plus has been the closer relationship and the emitting customer awareness to MIS. This has been an invaluable exercise and one where many skills can be easily transferred across the boundaries.

2.5.6 Truly applying technology as a weapon

This was a bold goal of truly harnessing technology to achieve better service. We have been successful thus far in applying information technology to helping our customers and we hope to continue to do so in the future.

3.0 In Conclusion

Unifying management responsibilities for MIS and Customer Service has worked well in our company. We have seen many discernible benefits, both to staff and the company.

The role of MIS is certainly evolving into a more generalist area through their comprehensive knowledge of the business, and it is this trend that should awaken IS Professionals to the tremendous opportunities that lay before them in other areas outside of MIS.

SELECTING AN AUTOMATED PROJECT MANAGEMENT SYSTEM

John F. Nichols
Nichols and Company, Inc.

Selecting, and eventually purchasing, an automated project management system is often a frustrating and sometimes a wasteful process. Most people know astonishingly little about the disciplines involved in project management.

Project management is like an unwanted child; most are quick to lament its condition, yet few resolve to do something about it. Rarely can any group justify the cost of building its own project management system. There are far too many good systems that can be purchased for a fraction of what it would take to design, build and maintain a workable product. Selecting an automated project management system is a process of knowing what you want, researching your choices and choosing the product that best fits your needs. It helps if this process is planned. You can start by analyzing who, within your organization, wants a system and why; try to understand why systems frequently fail to give the expected results; evaluate the organization's unique environment - it is important to understand the factors in your organization that will affect the use of such a system; establish particular and unique needs; and organize a logical selection process. Selection criteria should be established, and the products considered should be verified against the claims of what they can do.

WHY AN AUTOMATED SYSTEM?

People automate their project management efforts for many reasons. Some motives ease implementation, while others are sure to cause trouble. Grounds for automation should therefore be realistic and applicable and the need must be recognized by those who are responsible for its results: those individuals who must make it work, must have a reason for making it work. Some reasons for automation may be that the company's auditors want it, management wants it, project leaders want it, schedules are being missed, budgets are being missed or project leaders may recognize that their staff can be used more effectively.

Auditors have a particular perspective on how things work. They are usually quick to see certain signs of confusion and deterioration of control. These signs are often clearer to the objective observer than to the observed. The observed may not, however, share the auditor's view. The observed may see the problem as a shortage of people, materials or equipment. Whereas the problem may be the significance, utilization and coordination of those resources.

General management may be so caught up in the day-to-day business of running its designated areas, that it misses obvious signs of project deterioration. It too may have the insight to see that things are not right, but it may lack the tools to collect the right information. Its requests for information receive contradictory, incoherent and overly optimistic responses.

Project managers may find themselves unable to convince general management of the scope of its problems, or the need for additional capacity. General management may not perceive its views as contradictory, incoherent and optimistic: it may believe it needs more resources to do the job.

A consistent history of missed budgets or schedules in an organization may generate a universal desire to obtain some mechanism to bring a new sense of order. In that case, people approach the plan directly, rather than deal with misinformation. Often, when schedules and budget are missed, a new appreciation of the discipline appears.

There is then the rare group, or individuals, who recognize that project management is a strong and unique tool to better utilize people, and that with the proper application of a good system, people will know what they need to do next, waste will be reduced, and closer supervision can be maintained.

Sometimes, people look for a project management system to prove something to someone; they may end up proving themselves wrong. Project management should be seen as a positive tool for communication, evaluation, and determining perspective, not as a weapon or as a monument.

FAILURE OF AUTOMATED SYSTEMS

There are several reasons why so many automated project management systems may fail to live up to expectations. Expectations may be too high; or, in proving a point, the system proves the wrong point. There is a group of fundamental difficulties that frequently plagues the use of these systems. Rarely do any of these difficulties occur alone; most often, they accompany each other.

Probably the most common difficulty is simply a lack of group discipline. Each individual is allowed too much liberty as to whether, or how much, he will use the system. Errors are allowed to remain uncorrected, updates are random or sloppy, and other meaningless or contradictory methods are allowed to remain. Learning a system and its required discipline takes time. If implementation is rushed, this process cannot proceed properly. When an automated system is used with confusion, lack of understanding and simple ignorance, there is sure to be trouble and eventual failure.

Leadership is essential to a successful system. Management's lack of involvement can destroy even the best of implementations. All too often, management refuses to understand or participate in the use of a project management system.

Automated project management systems vary widely in complexity, documentation and applicability. Complexity can make a project management system difficult to use. It can force people with less experience or endurance to seek other methods or to give up entirely. A system's complexity arises from errors in design, or uncontrolled growth of features. Some products add a new input record for each new feature. Some products spread essential information out over many reports, destroying their use and ultimate value. Sometimes the essential information does not exist within the system.

Not uncommon, especially among new products, is an alarming lack of documentation. This creates the same symptoms as too much complexity. Information on how to use the system must be accessible, easy to understand, and must have consistent, reliable examples.

Finally, the product may not be suitable to the application. Some systems are highly application-dependent. If the applications require resource management and the product lacks these features, there can be little hope for success.

ENVIRONMENT EVALUATION

Having established clear goals for the automated systems, and having gained knowledge of why these systems sometimes fail in perspective, you can examine the environment to which the new system must adjust. There are some fundamental differences in environment that must be reviewed carefully before launching into purchasing a system. Probably the most fundamental of these is multiproject versus single-project management. If a group is attempting to manage many competing projects, it needs a system that stresses priorities and resource management. If it is a single project, then network and schedule management become key features.

Closely associated with this distinction is the way in which projects are controlled: is it project leaders or the staff planner who determines how much has been accomplished and what remains, or is each person held responsible for notifying the system what they have accomplished and what they have left to do?

The cumulative experience of the group that is assigned to use the automated system is also important. If there have been installed systems that were not successful, the group is frequently more aware of what its part will be in implementing a new system, and it can accommodate the new concepts more readily. Otherwise, the vendor may have to provide a great deal of basic schooling.

You must also consider the style of management. A nonassertive management may give people various tools without requiring their use. In this case, a well-integrated system would not be necessary, and summary reports are of little significance.

If it is expected that the project group will resist any automated system, intensive support may be required. Vendor support may verge on the situation of the vendor doing the work itself and setting an example. A vendor may be needed that gets highly involved and assists extensively in doing the work.

SPECIFYING MANAGEMENT NEEDS

Determining specific needs is the next step in the process of buying any automated project management system. Systems of this sort are becoming increasingly sophisticated with more and more modules and features, many of which are useless to most purchasers. Therefore, an awareness of these features is important. Often, the software an organization purchases has either far more or considerably less than is needed. Balance and forethought are essential in making the appropriate decision. The elements that should be examined fall into eight major groupings: ease of entry, ease of retrieval, PERT/CPM, resource management, progress reporting, performance analysis, accounting and product capabilities. Evaluators need to be aware of these categories and must appreciate fully the significance of each.

Ease of entry

Entry of information into any system is critical to the system's performance and eventual acceptance. Complex or convoluted screens and record formats discourage and confuse users, usually resulting in lower productivity, more errors, and resistance. Convenient, simple formats create an easy, workable environment. When people can understand formats quickly, they tend to be more accepting. Meaningful headings, a minimum of fields, a logical sequence of data and the elimination of nonessential data facilitate understanding.

In addition, there should be simple, convenient instructions for entry of the data via these formats. If the instructions are clear, the formats become easier to use. The instructions should be logical, have many consistent meaningful examples, and be simple and readable.

All nonessential formats should be eliminated: if the data is already present, or can be derived from data that is already present, it should not be re-entered. Poorly-designed systems have a high proliferation of formats resulting from afterthoughts. These additional formats may appear to give the system more power and flexibility, but they generally add to the confusion.

A system should not only edit all entries, but should validate them against the master file. All possible errors should be apprehended before processing. Extra work resulting from errors is time-consuming, frustrating, and tends to demoralize the user.

Security is a vital part of correct entry, and any entry method should therefore provide a thorough, integrated security mechanism. When uncontrolled entry is allowed, files are compromised and problems multiply. Security keeps unwanted data from contaminating the reliability of the system.

The ability of the system to provide files in a form that can be easily accessed by generalized report writers is not only convenient, but essential.

To simplify the overall planning process, planners should be able to copy easily significant information from other existing projects or project models. Project model copying can dramatically reduce planning effort and network errors. It can also provide a first-draft estimate on complex but repetitive work.

Ease of entry requires that the system be designed to accommodate the data from other related systems, such as accounting and personnel. Selective viewing and report distribution are necessary to reduce both compromising confidential files and the intimidation caused by excess information. As in entry, when uncontrolled access is allowed, positions are compromised and crossover responsibility occurs.

Ease of retrieval

Meaningful, easily-understood reports are as important as easy entry. Again, complex or convoluted screens or report formats discourage and confuse users. Poor retrieval lowers productivity, increases the chance for misinterpretation and builds considerable resistance among recipients of the reports.

Reports, whether online or printed, should be clear, concise and significant. Data should be sequenced meaningfully to facilitate understanding. To evaluate the use of certain reports, one can simulate different project-related situations, and see how many reports it takes to get an answer. This process is more easily accomplished by experienced project leaders and managers. The detail in reports must be adjustable. This is to say, the recipient should determine the amount of information a report contains. A voluminous report with extraneous information is intimidating. A poorly-designed system often produces miscellaneous information and forces the recipients to sort what they want from what they see. A well-defined system allows the person requesting a report to specify the extent and content, so that time and energy is not wasted scanning pages of unwanted or unneeded data.

Graphic reporting provides an efficient method for arranging and presenting information for quick interpretation. Graphic reporting improves management involvement and creates a wider, more enthusiastic acceptance of the system. Graphics should present both relational and quantitative information. Relational networks indicate the connection between tasks. Quantitative information shows values in either pie or bar format.

PERT and critical path method

PERT/CPM is generally not given the significance or credibility it deserves. PERT, which usually applies to time-estimating approaches to task management, has become a generic term for task networking. Simply stated, networking indicates the relationship of one task to another: it indicates the sequence of work, what work can be done concurrently and what type of overlapping is needed. Critical path method is a means of indicating what set of tasks, when connected, represents the longest path through a project. It is that sequence of tasks to which the addition of one day on any one task would add one day to the length of the entire project. Today, PERT/CPM embodies a multiple concept that might be simply referred to as task networking.

There are two basic forms of task networking: I/J and precedence. Precedence is easier to use and can express a wider range of variations, while I/J is still very popular among people with many years' investment in large scale projects. The most significant advantage with I/J is that it is much easier to express graphically, while precedence notation is easier to understand, making it easier to teach and making project presentation more accurate. To display precedence graphically, it should be translated into I/J.

Critical path analysis can have a dramatic effect on reducing the overall length of a project. It allows the planners to concentrate on the significant tasks that may lengthen a project unnecessarily. By concentrating on the critical path tasks, a planner can detect the need for parallel efforts, the need for additional capacity, or constraints due to outside factors.

Analysis of interproject dependencies is an important tool for the project planner. Systems that do not have the ability to connect tasks in one project to tasks in another project enforce the construction of larger and larger projects, to encompass naturally-related efforts. The size of any project representation is arbitrary: any large project can be composed of many smaller representations, while many small projects can have a single large representation. It is generally easier to manage many smaller representations. Thus, larger projects can be understood and managed easier if they are reduced to a series of smaller representations. This can only be done if the system has the ability to connect tasks between projects. Tasks do not always fit

conveniently end-to-end. That is to say, the next task does not always start when one or more other tasks are complete. There is a large number of possible variations when connecting tasks, or when showing the relation of one task to another.

Today, graphics representation is very significant. Network plotting, if required by management, should be an integral part of any product. The network displayed on the plot should be easily understood and have a time relationship.

Loop detection in PERT/CPM is worth mentioning. If there is a logical set of dependencies which end up depending on themselves, then the system should detect this, and present it in an easily workable form.

Resource management

Resource management is the discipline of optimizing the use of resources. Up to 15% of personnel costs is wasted because of the misuse of people. People waste time and effort waiting for work, working on the wrong assignments, reworking a task because it was originally worked without proper preparation, allowing people to select their own task or to define the content of their own task and so forth. All this waste is a result of inadequate observation and communication by management. Management simply does not have sufficient information in the right format at the right time to know what is happening, and often it cannot manage properly the amount of information being provided to it. A well-designed project management system can go a long way in assisting project management in this requirement.

Load levelling is one important service a project management system can provide for its users. It is the process of fitting an individual's or group's capacity to the assigned work. Once the capacity of an individual over some period of time has been used up, future assignments are moved to another time period where the individual has time available to work on the assignment. This movement is usually made forward in time and can demonstrate project delays due to lack of available capacity. Daily load levelling is significantly more useable and reliable than weekly or monthly levelling. This is because two days delay on a single task might mean delays of weeks or months in dependent tasks.

To make loading more reliable, the system should have the ability to load different daily patterns. A task, such as a meeting, may require just two hours a week but the two hours may have to always occur on the same day. A good project management system must accurately report the network and resource constraints such a task would create. For most tasks, there is a period of time when the task could start if the person or people necessary to work on it were available. This period is referred to as lag or resource delay. In other words, what is the

impact of the project if the right person or skill was not available when needed? A well-designed project management system should clearly highlight lag or delay along the critical path so that the planner may take appropriate action. This is referred to as resource availability impact analysis.

Skill balancing is an extension of impact analysis. It is possible to turn the project by arbitrarily increasing or decreasing the number of people with certain skills. This ability can have a dramatic effect on the cost of a project and the possibility of the project meeting its schedule deadlines.

A system with automatic assignment can optimize the assignment of people by comparing their individual skills to those required in the project. Automatic assignment should be continuous until an acceptable plan is finally produced. The system should continually adjust assignment as tasks are added, modified or removed. In this way, the system dramatically assists in the impact analysis and identifying the skills that are in short supply.

Progress reporting

The ability to report progress to a plan, and the manipulation of that progress, distinguish project planning systems from project management systems. Project planning systems either do not indicate progress or they use percent-complete analysis. Project management systems allow the people doing the work to report what has been done and what remains to be done. In that way, the system can more accurately recalculate the expected finish of the task and its impact on related efforts.

A second aspect of progress reporting is communication of the plan at every level, so that assigned people know what they are expected to do. This frequently takes the form of turnaround time sheet. This mechanism assures regular progress updates and keeps those doing the work advised on what has to be done. When notifying assigned people of what they are expected to do, daily itemization of work is useful. Daily itemization consists of notifying the person or group of what is expected of them during each day of the coming period.

Based on what has been done and what remains to be done, the system should recalculate future loading and the resulting schedule changes. This information becomes the basis for rescheduling the staff. When the work accomplished on a project is fully and regularly reported to the system, and individual estimates on what remains task-by-task are provided, the system can use a series of calculations to determine percent-complete. These calculations vary but are usually significantly more reliable than the sheer imagination of project management.

All resources should be considered in terms of progress. Resources not only include labor, but also materials and money. The appropriate people should be able to audit all resource consumption, whether labor, material, or money on a very detailed basis. This accountability is essential to project assessments.

Performance analysis

Performance analysis involves comparison of expected results to actual. There are three estimates to be evaluated: the original estimate, the revised estimate, and the in-process prediction. Each provides a guide for evaluation of relative effectiveness. Effectiveness of the estimate is distinct from the effectiveness of the assignee to accomplish the work; although both are determined by a series of comparisons of similar situations.

Comparative analysis must first look to its resource and schedule estimate for the project as a whole and then to the parts of the project such as phases, activities, tasks and so forth. It must also compare types of work, assignees, accounting categories and skills; how does each do when set alongside similar items? From performance information comes utilization analysis, a review of how much was produced based on the type of work done, the individual doing it, and the skills they possess. This information becomes a basis for improvement.

Accounting

Accounting is similar to performance analysis, except that accounting uses a single measure of performance: that is, money. If the project management system is well-designed, it allows project management to be well-advised of the expenditure of all sums in its various areas of accounting. Without deleting information from the file, it should be possible to establish, for each project, the phase, activity and task effort to date, year to date and month to date itemization.

To provide this for the widest range of needs, expenditures should be presented by any categories contained within the system. That is, expenditure should be categorized by individual, group, department, type of work, work breakdown structure, project, phase, activity, task, skill or any other reference available. When expenditures are categorized, they should be in increments of week, bi-week, month, bi-month, quarter, or year.

The general ledger is an important aspect of any business. The general ledger system should be able to receive data directly from the project management system without a second translation or manual interface. To do this, all relevant general ledger categories should be applied in the project management system before information is passed to the general ledger system.

Customer billing is a natural by-product of project management. The project management system should not only provide for it, but allow for all areas of resource expenditure to be covered in the billing operation. It also should be flexible enough for all the users to control the billing increment.

Capacity is a frequently overlooked area of analysis. It is an area that requires some thought. The first consideration is whether the system's accounting facility has significant limitations, and if it does, what they are. A list of items that should be questioned is: projects, phases, activities, tasks, dependencies, resources, types of work, assignees, skills, accounting codes and categories, how far into the future the system can schedule, and how much past information the system can store and display.

DEVELOPMENT OF A SELECTION PROCESS

At this point, you should know why an automated project management system is wanted, why it may fail, what the environment is like and what the specific needs are. Armed with this insight, a selection process can be developed. The selection process is the final crucial step. If it is not done well, then all the preparation is of little value. If it is done well, the chances are that the chosen product will be the correct one for its application and will work in its environment.

In any evaluation, it is the evaluators that make the difference. It is therefore important that they are chosen carefully. A team of no more than three people should be selected. These people should have a background in general management, project managing and in working with automated systems. They should have a 'show-me' attitude. Because it is a management system they are investigating, the team leader should be the one with the strongest management orientation. The time-frame for the selection process should be 60 days at about one-quarter effort for each person.

The team members should decide on the initial criteria. They should interview potential users, attend seminars, and generally gather all the preliminary information they can. This list of criteria should be broken into essential items, useful items and luxury items. The criteria should be reviewed on several occasions with all levels of management on the basis of its validity, and should be discussed and constantly updated. The criteria should be grouped as discussed earlier into: ease of entry, ease of retrieval, networking, resource management, progress reporting, performance analysis, accounting and capacities. Added to this list should be items concerning the vendor such as: willingness to assist, flexibility, proficiency in the subject and reliability.

The most viable vendors should be determined as quickly as possible. Each vendor should be asked to make a presentation on-site over a relatively short, perhaps ten-day, period. Detailed notes on each presentation should be kept and, if possible, a tape recording of the speaker should be made. If the sales person does not have an answer for some questions, make a note and follow it up. All significant questions should be answered within a few days. Each potential vendor should be asked to respond on the reliance of the criteria and asked if they have possible additions.

From the presentation, all but two or three products should be eliminated with an explanation of why. Of the two or three remaining, references should be contacted with very specific questions prepared in advance. If this short list still stands, then visits should be made to the vendor's in-house marketing seminars to gain hands-on experience.

From all the information gathered to this point, a final review should be made to determine the best possible candidate. The final candidate should be willing to provide its product for a 30-day trial. Everything should be done to prepare for the trial, so time is not wasted. Projects should be prepared in advance, working solely with the vendor's manual. Vendor assistance during the trial should also be evaluated carefully.

The trial should be conducted by the evaluation team. This is not the time to get all interested people involved, because, for the most part, they do not know what they are looking for. This is also not the occasion to decide whether one wants project management; that should have been decided long ago. This time is only to evaluate the product and whether it does what is expected. If there are problems, the vendor should be given resonable time to respond to questions or to make corrections. If the tests verify the product, it should be bought; if not, the next product should be tried.

When this method is used to choose a project management system, the chances of getting just the right system for your organization will be optimized. You will have chosen the system that best addresses expectations. You will have identified your organization's specified requirements, studied the products and features available to meet those requirements, and will have assured yourself that the product you chose properly serves those requirements.

When the time comes to install the new system, the orderly manner in which the system has been chosen will dictate the course of installation and implementation. All levels of management will have been involved, thus paving the way for greater acceptance. The organization's size, style, and expertise will have been considered, and from this a vendor will have been found that is eager to assist in every step of installation and implementation.

In short, reliable results always depend on a well thought out, carefully executed plan.

Nichols & Co., Inc. 5839 Green Valley Circle, Suite 104, Culver City, CA 90230, USA

This paper was first published in the proceedings of the 1985 Internet World Congress, Rotterdam, May 1985 - North Holland 1985.

System Performance: A Case Study

System Performance: A case study

By: James S. Harvison, Vice President
Charles Coleman, Programming Mgr.
Computing & Telecommunications Services
M. S. Ginn Company, Hyattsville Md.
301 - 853 4397

Computing & Telecommunication Services 1

A means to an end

System Performance is a journey rather than a destination.

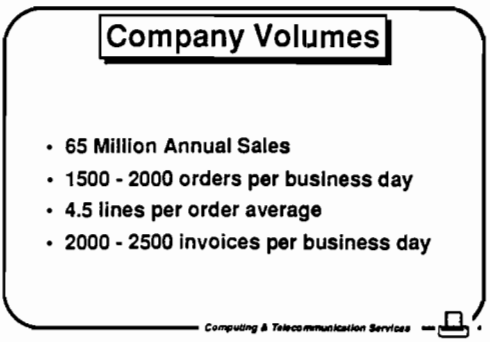
Factors:
Environment Internal & External
- Hardware
  - New and faster equipment choices
- Software
  - Performance Enhancers
  - Performance Detractors

Computing & Telecommunication Services 2

3031-1

Presented By: James S. Harvison; Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

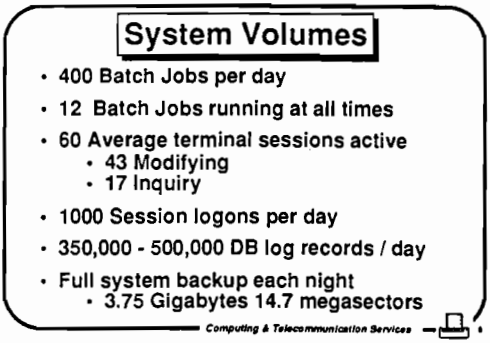
System Performance: A Case Study

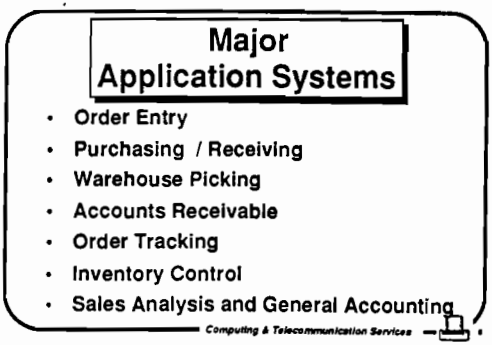


3031-2

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

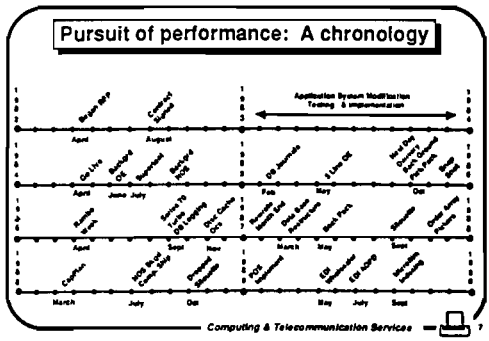
System Performance: A Case Study

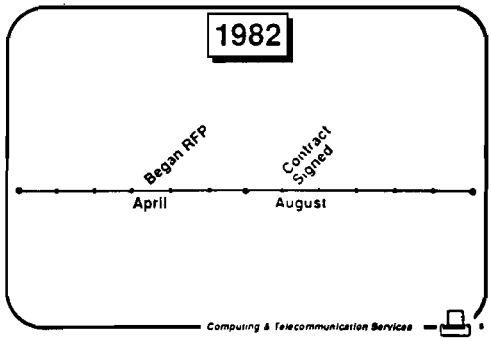




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

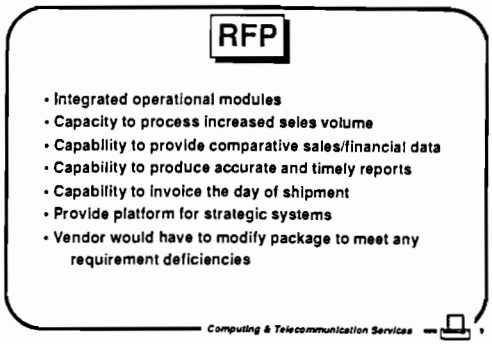


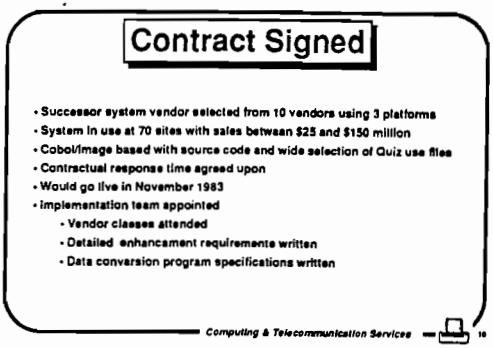


3031-4

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

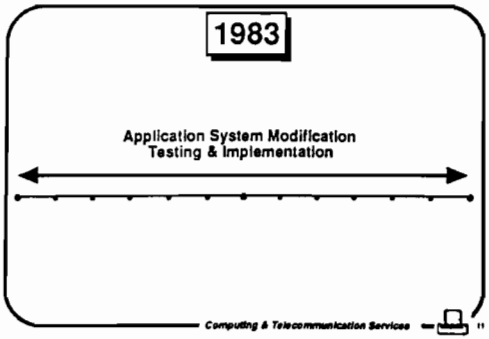


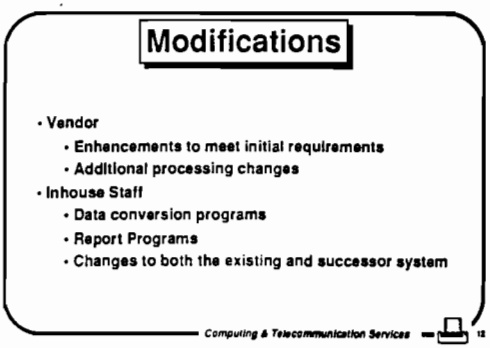


3031-5

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

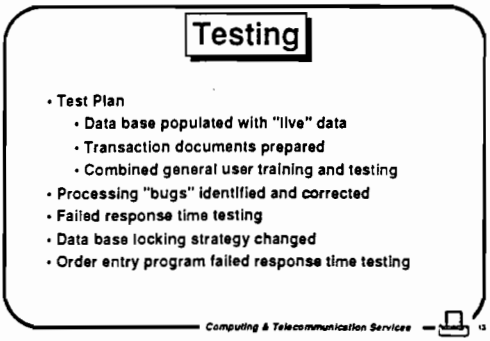
System Performance: A Case Study

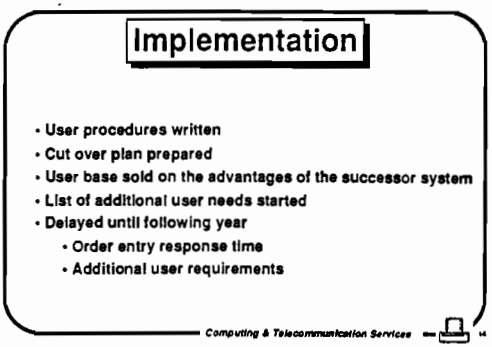




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

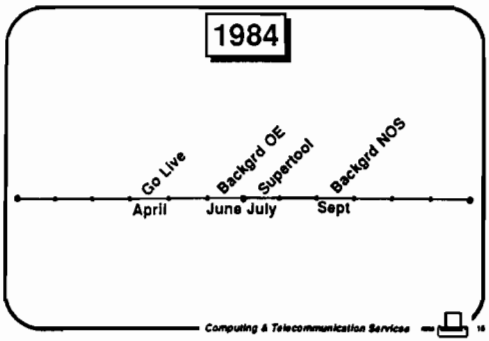
System Performance: A Case Study





Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study



Going Live

- Before
  - Order entry program met response time test
  - Essential user requirements were met
  - User expectations were high
  - Customer invoicing was six months behind
- After
  - User response time
  - Excessive run times
    - Reports
    - Data archiving and deletion
    - Month end processing
  - System Backups
  - Limited system availability

Computing & Telecommunication Services

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

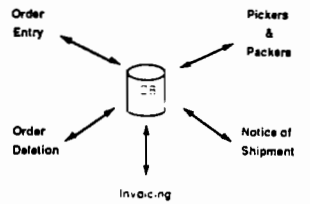
System Performance: A Case Study

Decision Time

- Expectations not met
  - Reports not available
  - Growth capacity not evident
  - Customer invoice backlog growing
- Alternatives
  - Roll back to the old system
  - Replace the software and possibly hardware
  - Rewrite the system

Computing & Telecommunication Services 17

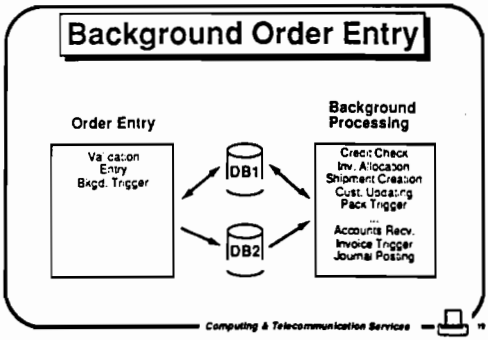
Order Cycle

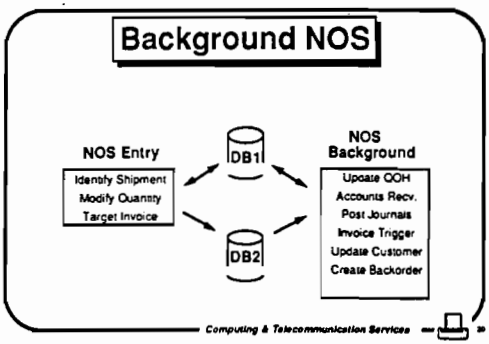


Computing & Telecommunication Services 18

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

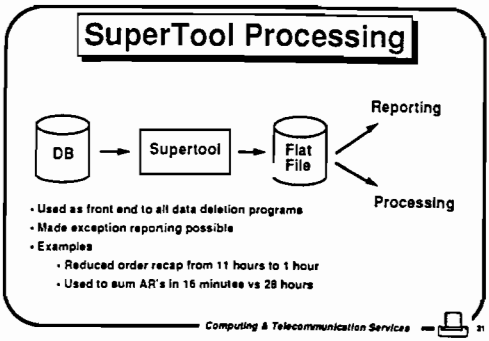


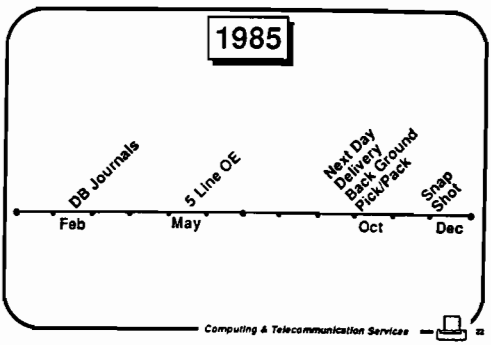


3031-10

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

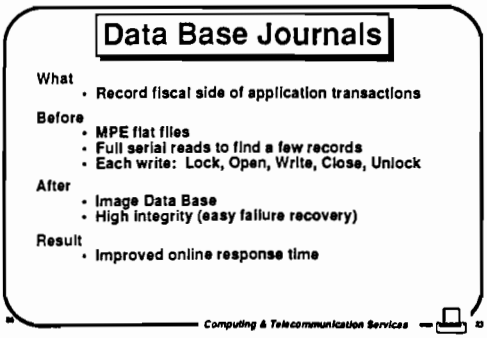


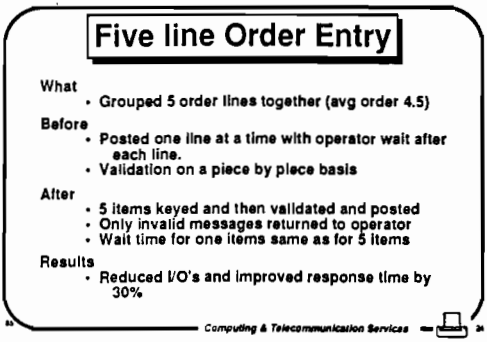


3031-11

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

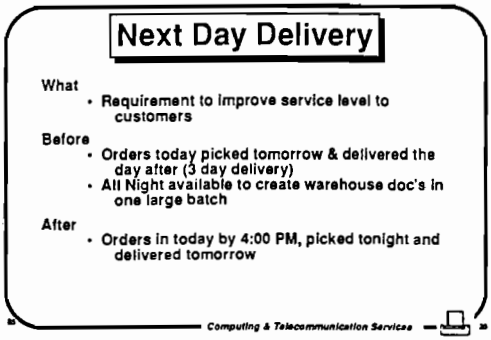
System Performance: A Case Study

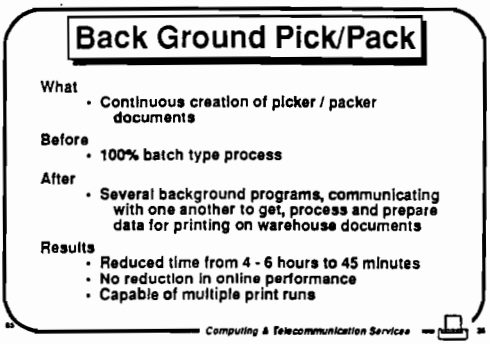




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

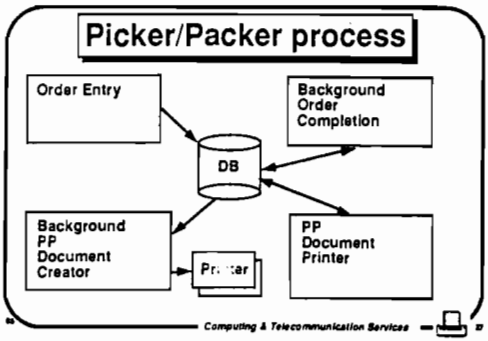
System Performance: A Case Study





Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study



SnapShot

What

- 2 Hour data collection of system performance during prime period on Series 68

Findings

• CPU Busy	92.1%	• CPU Wait	7.8%
• User	43.9%	• Disc Swap	0.3%
• MPE	20.6%	• User Disc	6.9%
• Cache	27.6%	• Idle	0.6%

- No memory pressure (7 megabytes)
- Physical I/O @ 36.8 per second
- User Disc Caching 81.8% hit
- I/O balanced across the 6 drives

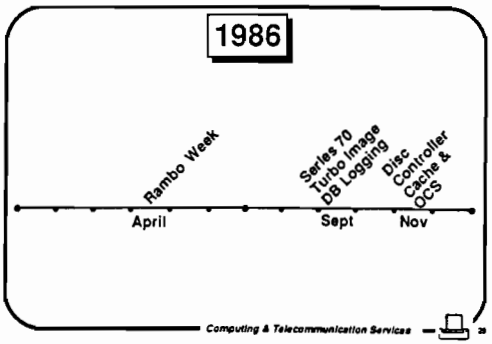
Recommendations

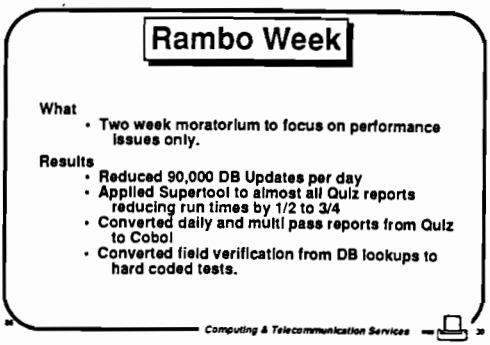
- Allocate programs
- Install modem signal patch to reduce TT10 opens
- Optimize Disc I/O's thru file relocation

Computing & Telecommunication Services

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

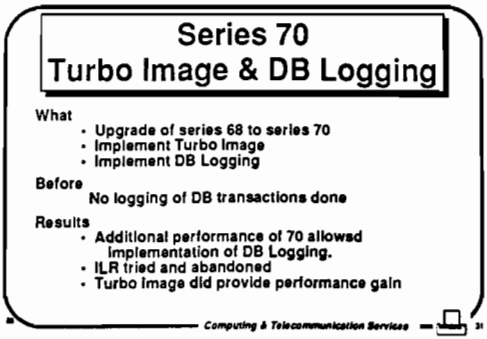


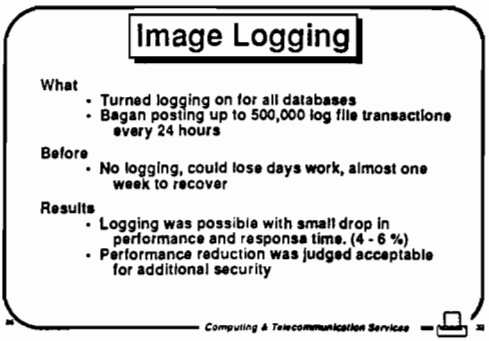


3031-15

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

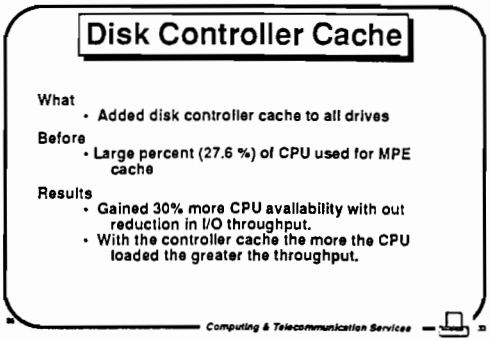
System Performance: A Case Study

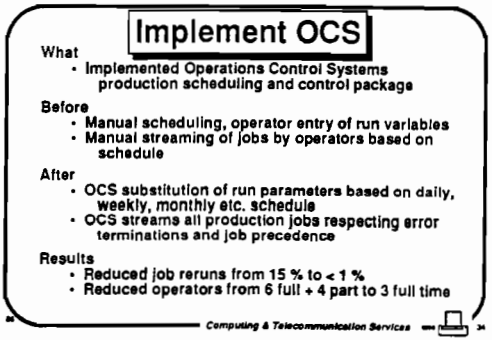




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

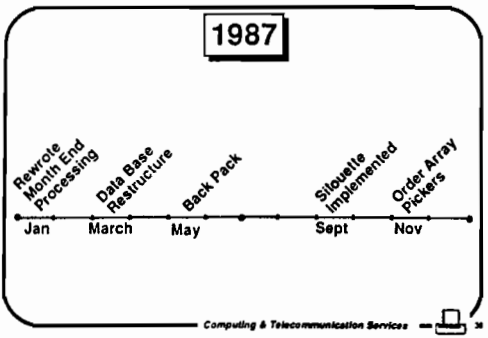
System Performance: A Case Study

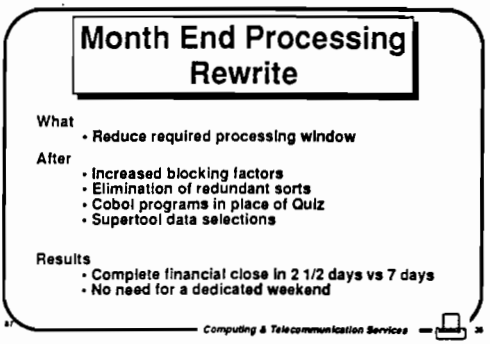




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

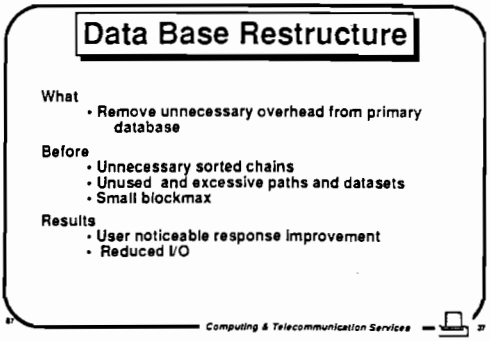


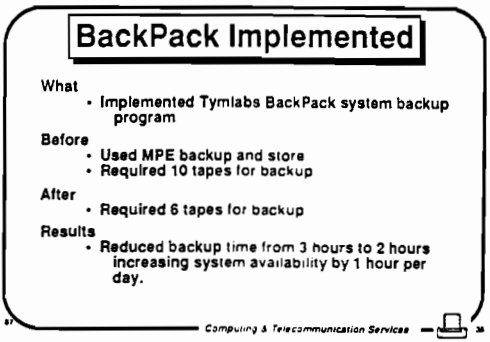


3031-18

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

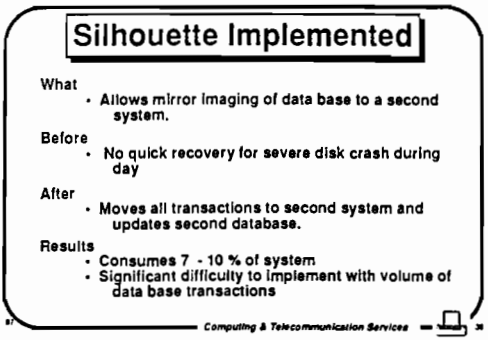
System Performance: A Case Study

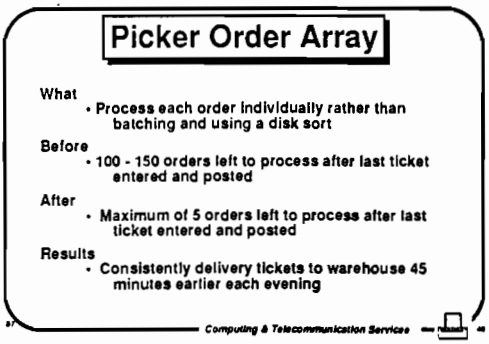




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

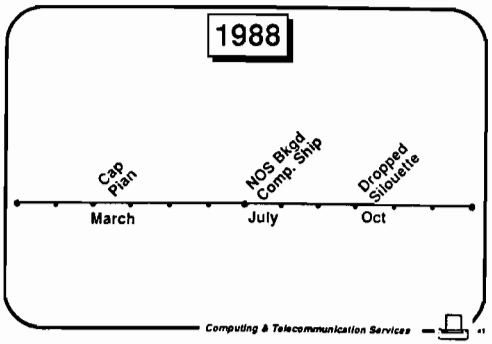
System Performance: A Case Study

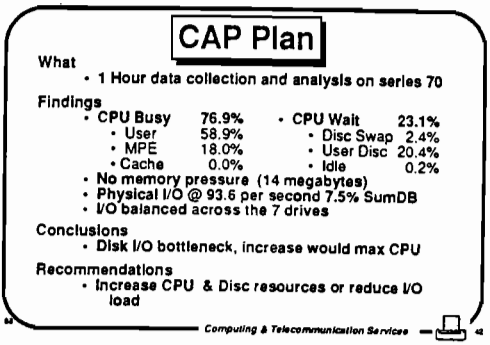




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

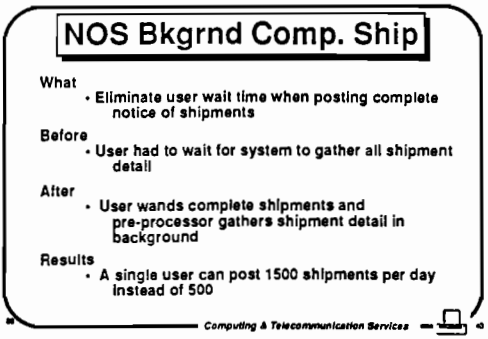


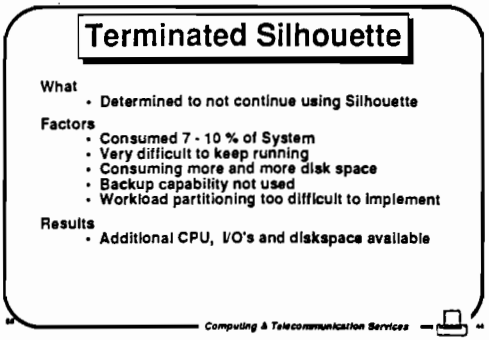


3031-21

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

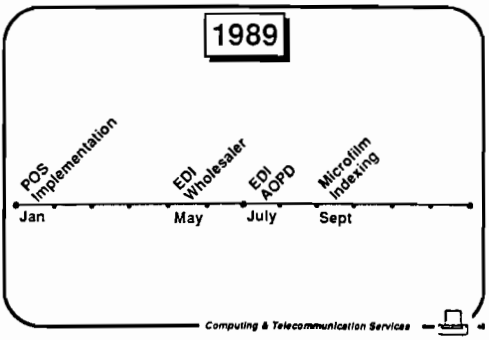
System Performance: A Case Study

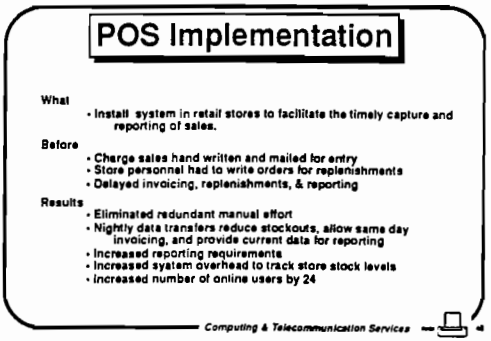




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

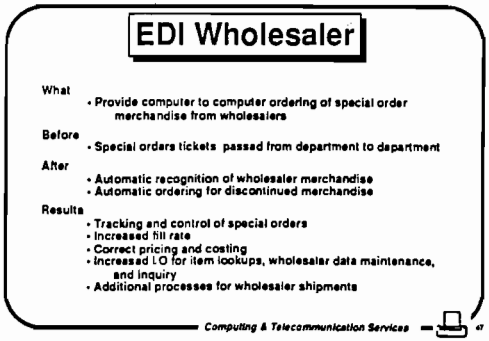


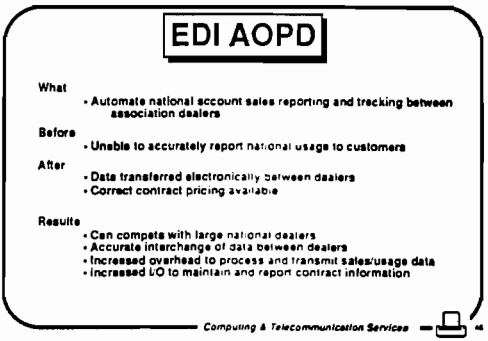


3031-23

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study

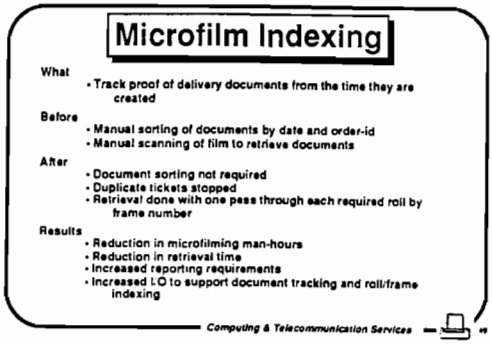


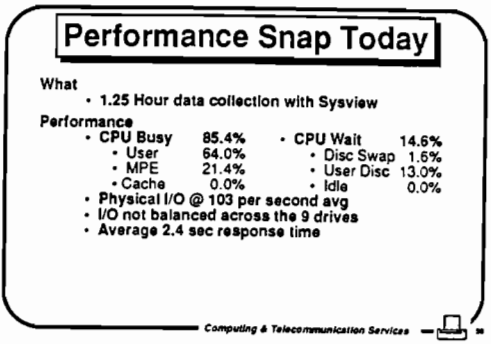


3031-24

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

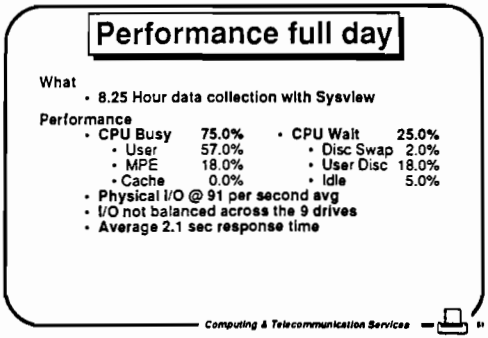
System Performance: A Case Study

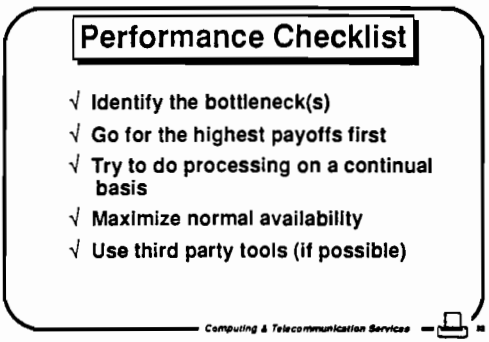




Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

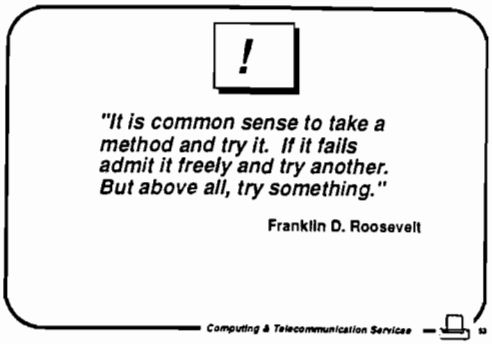
System Performance: A Case Study





Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

System Performance: A Case Study



3031-27

Presented By: James S. Harvison, Vice President
Charles Coleman, Programming Manager
Computing & Telecommunications Services
M. S. Ginn Company, 5620 Ager Rd. Hyattsville, Md. 20782
(301) 853-4397

TITLE: Turning Promises Into Realities: Formulating a Successful OA/DSS Strategy

AUTHOR: Michael Levinger (508) 655-9191
Access Technology, Inc.
Two Natwick Executive Park
Natwick, MA 01760

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 4003

SELECTING AND MONITORING LONG DISTANCE SERVICE

Richard Kasper
Capitol Health Care
201 High St SE
P.O. Box 12625
Salem OR 97309

DISCLAIMER CONCERNING ACCURACY OF RATES AND SERVICES

All rates and services used in this paper should be considered illustrative only. Long distance rates and services are changing daily, and may have changed between the time this paper was printed and the time you begin to use them. Check with the specific vendor before using any of the rates or services presented in this paper. Neither Capitol Health Care nor the author of this paper assumes any responsibility for the accuracy of rates or service information presented in this paper.

I. INTRODUCTION

LONG DISTANCE HISTORY

In order to understand where we are today in regards to long distance service options, we need to understand what the past was like. I know that there are some days that I long for the time when the only option was AT&T. Some times it seems that a good system was fixed before it was broken. But what happened to the phone company?

Like any large corporation, there have always been companies that resented AT&T's competitive practices and wished to compete in the same markets. Concern about AT&T's size and monopoly power prompted the Justice Department to bring an antitrust suit against AT&T as early as 1949. This was settled in 1956 with the ruling that prohibited AT&T from entering any business other than regulated telephone services. This prevented AT&T from competing in the rapidly going computer industry.

Subsequent to the 1956 consent decree, the major milestones have been:

1968 - FCC rules that customers can connect any non-Bell equipment to their phone systems as long as it is compatible.
1969 - MCI was granted permission by the FCC to build a microwave transmission system between Chicago and St.

Louis. This allowed MCI to offer long distance service at a lower rate than AT&T.

1974 - The Justice Department files suit against AT&T charging them with freezing out the competition in the long-distance and equipment markets by using the Bell companies.

1978 - a federal appeals court rules against AT&T requiring them to allow long-distance companies to connect at both ends of a call in order to provide dial-up service. This lead other companies to join MCI in competing for long distance service.

1980 - AT&T is allowed to set up a subsidiary, initially named American Bell and later changed to AT&T Information Systems. This allowed AT&T to compete in the markets for telephone and many computer related services but only as a fully separate subsidiary. This meant that monopoly profits could not be used to subsidize sales in competitive markets.

1981 - FCC allows unlimited resale of all interstate telephone service. Anyone can buy WATS service and resell that service at rates that make a profit for the reseller while undercutting the rates available from AT&T on a call by call basis.

1981 - Judge Harold Greene refused to dismiss the case against AT&T.

1982 - AT&T and the Justice Department settle the antitrust suit. AT&T agreed to give up its local operating companies in exchange for the freedom to compete in other unregulated markets. Later in the same year Judge Greene signed a modified agreement and AT&T filed a reorganization plan which provided for the division of the assets and personnel of the 22 Bell operating companies (BOC).

1983 - Judge Green approved the plan.

1984 - AT&T gave up ownership of the operating companies.

1986 - Equal access went into effect.

AT&T is now quite different from the way it was. It no longer holds an interest in any of the Bell operating companies. It lost its name and logo, the Yellow pages and revenues from local service and some intrastate toll calls.

The local BOCs have become subsidiaries of seven regional companies. These regional companies receive revenue from all local service, some intrastate toll service and the Yellow pages

WHY MONITOR COSTS

The cost of long distance services has changed dramatically in the past couple of years. Many companies have not taken the time to examine the new services being offered and as a result may be paying more and getting less for the dollars

they are paying. It is the responsibility of telecommunications managers to reevaluate their long distance service choices occasionally to see if a substantial savings can be made by switching services. Most companies are paying from $2,000 to $3,000 a month or more for long distance services. This gives the telecommunications manager the opportunity to add to the bottom line by reducing expenses and at the same time providing telephone service which will effect the service given to customers and could mean increased revenues from sales generated by improved service.

OVERVIEW OF PAPER

This paper will help the telecommunications manager make a better evaluation of long distance services by offering explanations of what services are available and by offering a process for evaluating the various services. Once selection of long distance services is made, it is then necessary to monitor the quality and cost of those services. This paper will present ways to conduct this monitoring. Judicious application of techniques presented in this paper will help reduce a companies long distance telecommunications costs.

II. CHARACTERISTICS OF LONG DISTANCE SERVICES

TYPES OF SERVICES

DIRECT DISTANCE DIALING
The most often used and most expensive long distance service is dialing directly using a selected carrier. This type of service is called Direct Distance Dialing or DDD and sometimes referred to as "dial-one" service. With the advent of equal access, we now have the option who our DDD long distance carrier will be. When a interLATA long distance call is placed, it will automatically be routed to the carrier you have selected. A LATA is a geographical area that was established by the Justice Department that defined the boundaries that the telephone companies would be permitted to provide long distance service for. It should be noted that LATAs do not always follow area codes or in some cases even state boundaries. Normally, intraLATA long distance calls will be carried by your local telephone company, even though they may not be your selected carrier. If you want your selected carrier to carry your intraLATA long distance calls, you must dial the special access code for that vendor. This could be done manually or be done by your PBX through least cost routing.

Direct Distance Dialing has the following characteristics:

1. A detail billing of each call is received.
2. There is a minimum length billed per call, usually one minute.
3. Calls are billed in one-minute increments.
4. First minute is billed at a higher rate than subsequent minutes.
5. Airline milage is used for billing.
6. Evening and night calls are discounted.

WATS (Wide Area Telephone Service)
There are four main types of WATS service offered by the major carriers. These are banded WATS, virtual WATS, dial-one WATS and T-1 WATS access. All of these are discounted long distance service.

Banded WATS
With banded WATS a dedicated line is installed from the your office to the long distance carrier. There is a fixed monthly service charge which includes the cost of the line. There are six WATS bands and when installation of the dedicated line is done a destination band must be chosen. The rate charged will be determined by the bands chosen. It is important to note that a call to a lower band area can be carried by a higher band line but calls for a higher band can not be carrier by a lower band line. For example, if you have a Band 2 and Band 5 service, the carrier would provide two dedicated lines. If the Band 2 line was full, calls for Band 1 and 2 could be carried by the Band 5 line. However, if the Band 5 line was busy, calls for Band 5 could not be routed over the Band 2 line.

Banded Wats has the following advantages:

1. It is less expensive when calls are frequently made to a single band.
2. Number of local trunks required would be less as dedicated lines would be used to handle the WATS calls.
3. Fixed costs are lower than other discounted services.
4. Calls are billed by hours or minutes of use. The higher the amount of use per month, the lower the rate.

Banded WATS has the following disadvantages:

1. Banded WATS does not allow for seasonal trends.
2. If usage is low and only requires one or two trunks, the inefficiencies of small trunk groups would be encountered.
3. Calls overflowing to higher band areas will be at higher rates.

Virtual Banded Wats
Just as with Banded WATS, Virtual WATS uses dedicated lines to the selected carrier. However, traffic is carried on the same lines to any of the WATS bands. The call charges are based on band destination but at a higher rate than for banded WATS.

The main characteristics of virtual WATS are:

1. Calls to any band share a common group of dedicated lines.
2. Calls are billed at a flat rate regardless of amount of monthly usage.
3. When all lines are full, the traffic is blocked or overflows to DDD.

Dial-one WATS
This service is similar to DDD except that a fixed monthly charge is made in exchange for a discounted rate structure on long distance calls.

The major characteristics of dial-one WATS are:

1. No dedicated lines are needed.
2. Service is identical to DDD except for fixed monthly rate.
3. Detail of calls shows on bill.
4. All calls are carried as long as there are enough trunks to the local telephone company.

T-1 WATS or Bulk-rated WATS
Most of the major carriers offer T-1 access to large users of long distance services. There is a fixed monthly cost for access but the rate for usage is lower because of the volume carried by the T-1 channels.

Major characteristics are:

1. Generally a high monthly usage is necessary to justify the fixed access charge.
2. The user must provide a way to split out the individual channels of the T-1.

INWATS OR 800 SERVICE

Usually, 800 service is viewed as a service allowing a person to place a call to a company at no expense to the calling party. However, it can be a cost saving tool for calls being placed into the home office by employees of the company, also. Branch offices to small to have an outgoing WATS service can reduce the cost of calls to the head office by using an 800 number. Likewise, sales persons on the road can use an 800

number to reduce the expense of calling for messages or placing orders. Dedicated trunks are used for 800 service and a fixed charge is incurred for the dedicated line with costs for the calls varying with the amount of usage.

OTHER LONG DISTANCE ALTERNATIVES

Foreign Exchange Service
With Foreign Exchange Service, a dedicated line is used to bring dial tone from one city to another. An FEX can be used to allow persons in another calling area to reach your offices by dialing a local number. FEX service can also be used access the local calling area of a distance city.

Remote Call Forwarding
When volumes are to low to justify a dedicated line, Remote Call Forwarding is a good alternative. This allows persons to dial a local number and have the call forwarded to a distance city. Charges for the call from the local number to the remote destination are billed to the called party. A similar feature called remote call transfer would allow a person to call a local office and have the call transferred to an office in another city.

Tie Lines
A Tie line is used to connect to PBXs in different locations. This allows access to extensions at either location and also access to DDD trunks at the remote location. When PBXs are tied together using a uniform numbering plan, the PBXs can function virtually as if they were one.

LONG DISTANCE SERVICE - WHAT DO THE VENDORS CALL THEM

The following chart is a comparison of the names given by AT&T, MCI and US SPRINT to the long distance services discussed in the previous section. In order to keep this paper from sounding like a sales pitch, specific information will not be given regarding these services. Please contact a sales representative from these carriers for any information you may need. It should also be noted that there are several resellers of long distance services. These vary from region to region. In some cases, these resellers will save money over the three major carriers whose services are listed in the chart below. Because of the brevity of this paper, however, these other vendors were not covered but it may be beneficial to compare the services of the resellers in your area before choosing a carrier.

	AT&T	MCI	US SPRINT
DDD	AT&T LONG DISTANCE	MCI DIAL "1"	DIAL "1"
BANDED WATS	WATS	MCI WATS	------
VIRTUAL WATS	------	PRISM II PRISM III	ADVANCE WATS ADVANCE PLUS
DIAL-1-WATS	PROWATS	PRISM PLUS	DIAL-1-WATS
BULK RATED WATS	MEGACOM MEGACOM 800	PRISM I	ULTRA WATS ULTRA 800
INWATS	BASIC 800 800 READYLINE	MCI 800	FONLINE 800

III. WHAT'S THE RIGHT CHOICE FOR YOUR COMPANY

The two major factors I use to evaluate the services offered by a carrier are cost and quality of the service. Both cost and quality are made up of several variables and so some means must be used to compare these factors from carrier to carrier.

Chart labeled "A Few Questions to Ask the Vendors, located at the end of this paper, can be used to record responses to various questions ask of the carriers under consideration. After the chart is completed, information can be compared and cost information can be extracted for further analysis.

A brief explanation of each question follows:

Is there a installation or setup fee?
This fee can vary from the $5.00 fee to switch carriers for direct dial long distance to several thousand dollars for the installation of a T-1 access. Installation or setup fees should be subtracted from first year savings to show the actual amount saved by switching to a new carrier.

Is there a monthly flat fee?
Most plans using dedicated lines will have a monthly fee. This also should be used as an offset to anticipated savings.

Is there a monthly usage requirement?
Some plans are oriented towards large volume users and have minimum usage charges. You may be charged for the difference between actual usage and minimum usage.

What are the volume discounts?
Most of the plans offer volume discounts which generally get larger as volumes increase. This discount may be applied to the whole bill or only to the portion that exceeds a specific amount.

What is the first minute rate and what is the subsequent rates? Most carriers charge more for the first minute and less for each subsequent time. This plan will favor those who make long calls versus those who make shorter calls.

Do rates vary with distance?
Rates for most carriers are set based on the distance of the call. This distance is usually airline mileage calculated based on the vertical and Horizonal (V & H) coordinates of the cities where the call originates and terminates. Most call accounting systems contain V & H coordinates for determining airline mileage to calculate rates for calls.

How are calls timed?
Some companies have to estimate the time when a call begins and when it ends. This is due to limitations of the connections between the local telephone company and the carrier. These carriers begin to bill after a predetermined number of seconds. This would mean if you allowed the phone to ring passed a certain number of seconds and still received no answer and then hung up, you would be charged just if you completed that call. If you begin to notice a large number of 6 second calls on your bill, it is most likely because the vendor is using software timing and is not able to detect the actual completion of the call. Almost all carriers round the call up to either the next tenth of a minute to the next full minute.

Who bills for long distance calls within my LATA?
Most intraLATA calls are billed by the local telephone company unless a special access code is used.

Now that you have information on features, rates and billing practices, you should be able to select a few carriers that you would like to do cost analysis on. The following case studies are examples of different ways to determine cost of using a selected service. Your own usage and rates can be substituted for those used in these case studies.

The spread sheet labeled "Calculating Virtual WATS Cost", located at the end of this paper, is used to calculate the total cost of long distance service obtained from two different services offered by the same vendor. The range is the mileage bands for the offered service. Typical mileage bands are represented in the following table:

Band	Mileage
1	0 - 55
2	56 - 292
3	293 - 430
4	431 - 925
5	926 - 1910
6	1911 - 3000
7	3001 - 4250
8	4251 - 5750

The minutes used is taken from current bills. The rates for the service are supplied by the vendor. The Busy Hour Erlangs is the amount of traffic in the peak hour of usage. The number of trunks required is calculated using traffic analysis methods which are beyond the scope of this paper. Usually a vendor will be willing to supply you with the trunks that would be required to carry the necessary traffic. This spreadsheet does not include installation costs for the dedicated trunks as this would be a one time charge. This charge should not be overlooked as it could be large enough to nullify any savings from the offered service.

Also included with this paper our proposals from two vendors. These proposals were done for Capitol Health Care and contain rates for our volume and own particular circumstances. These proposals are provided as examples of how two vendors presented their offered services and should not be used as a basis for a decision on the service to select.

IV. MONITORING SERVICE

Telecommunications charges are a large part of your company's expenses and should be checked regularly for accuracy. Costs should be analyzed periodically to help control cost and to adjust levels of service as necessary. Rates and services change frequently and the telecommunications manager has the responsibility to check bill and question fluctuations and changes in service costs.

Long distance bills should be analyze every month to detect trends. A spread sheet like the one labeled "Long Distance Summary" could be used to track trends. This spreadsheet can be converted easily into a graphic helping to see the up or down trend of cost and usage. Also included on this spread sheet are the action levels for each type of service. When cost or usage reach these levels, the telecommunications manager is alerted to the need to do a detailed analysis of the service.

V. CONCLUSION

Today's telecommunications manager has an excess of alternatives available today for long distance service. If managing the phone system is not your primary responsibility, it becomes difficult to stay abreast of all the changes. As one carrier changes his rates, the others will do so, too. When your vendor notifies you of a major rate restructuring or an addition of new products, you should study the changes well to see if a change is warranted. Many companies have failed to change, and as a result have spent more money than necessary to obtain adequate service.

A FEW QUESTIONS TO ASK THE VENDORS

NAME OF COMPANY	COMPANY 1	COMPANY 2	COMPANY 3
1. Is there an installation fee?			
2. Is there a monthly flat fee?			
3. Is there a monthly usage requirement?			
4. What are the volume discounts?			
5. What is the first minute rate and what is the subsequent rates?			
6. Do rates vary with distance?			
7. How are calls timed?			
8. Who bills for long distance calls within my LATA?			

CALCULATING VIRTUAL WATS COST

RANGE		MINUTES USED	PRISM PLUS RATES	PRISM PLUS USAGE COST	PRISM 3 RATES	PRISM 3 USAGE COST
	1	2347	0.1962	460.48	0.2107	494.51
	2	4431	0.2097	492.17	0.2185	512.82
	3	6543	0.2241	525.96	0.2308	541.69
	4	336	0.2313	542.86	0.2455	576.19
	5	8765	0.2448	574.55	0.2559	600.6
TOTAL		22422		2596.02		2725.81
BUSY HOUR ERLANGS		2.9				
DEDICATED TRUNKS		6			35	210
ADDL. C.O. TRUNKS		3	52	156		
TOTAL COST				2752.02		2935.81

AT&T PROPOSAL

I. CURRENT CONFIGURATION

A. OUTGOING TRAFFIC

ACCOUNT	BAND	LINES	HOURS	USAGE	ACCESS	TOTAL COST
503 3XX-XXXX	DDD	—	4.12	$83.11	—	$83.11
PRISM II		2	130.99	$1,148.20	$260.00	$1,408.20
PRISM III		2	33.23	$379.83	$70.00	$449.83
TOTAL			168.33	$1,611.14	$330.00	$1,941.14

COST PER MINUTE: 19.2 CENTS

B. INCOMING TRAFFIC ON 800 LINES

XXXXXXXXX	0	4	72.2	$965.56	$160.00	$1,125.56
XXXXXXXXX	0	2	18.6	$247.38	$80.00	$327.38
XXXXXXXXX	0	1	30.4	$381.22	$40.00	$421.22
TOTAL		7	123.2	$1,594.16	$280.00	$1,874.16

COST PER MINUTE: 25.4 CENTS

II. PROPOSED CONFIGURATION

A. OUT GOING TRAFFIC

1. AT&T MEGACOM WATS WITH OVERFLOW TO DIRECT DIAL

SERVICE	CHN	HOURS	USAGE	MONTHLY RECURRING	ACCESS	TOTAL COST
MEGACOM	6	164.21	$1,080.07	$50.00	T1.5	$1,130.07
DDD		4.12	$83.11			$83.11
TOTAL COST		168.33	$1,163.18	$50.00	T1.5	$1,213.18

COST PER MINUTE: 12.0 CENTS

B. INCOMING TRAFFIC

1. AT&T MEGACOM 800

SERVICE	CHN	HOURS	USAGE	MONTHLY RECURRING	ACCESS	TOTAL COST
MEGACOM 800	6	123.2	$912.86	$50.00	T1.5	$962.86

COST PER MINUTE: 13.0 CENTS

C. COMBINED TRAFFIC	CURRENT	PROPOSED
1. 1 T1.5	——	$316.39
2. OUTGOING	$1,874.16	$1,213.18
INCOMING	$1,874.16	$962.86
	$3,815.20	$2,492.43

PROPOSED OVERAL COST PER MINUTE: 14.5 CENTS

SAVINGS PER MONTH	$1,322.87	INSTALLATION CHARGES	T1.5	$2,344.58
SAVINGS PER YEAR	$15,874.44			$644.00
				$1,189.00
				$1,833.00

T1.5 CHARGES WAIVED IF ORDERED BY 8-10-89

OUTGOING SERVICE

CURRENT CONFIGURATION

SERVICE	HOURS	COST
U.S. West Long Distance	.3	$ 4.78
U.S. West Credit Card Fee		$ 5.00
U.S. West Credit Card Usage	.9	$ 18.19
AT&T Credit Card Usage	1.2	$ 22.17
U.S. Sprint Credit Card Usage	.03	$ 3.68
(2) MCI Prism II Access Lines		$ 260.00
MCI Prism II Usage	109.4	$ 1,013.34
(2) MCI Prism III Access Lines		$ 70.00
MCI Prism III Usage	17.1	$ 211.32
TOTAL CURRENT COST	128.93	$ 1,608.48

PROPOSED CONFIGURATION

T-1 ACCESS

SERVICE	HOURS	COST
(5) MCI Prism II Access Lines		$ 0.00
MCI Prism II Usage	126.8	$ 1,124.69
MCI Card Usage	2.13	$ 39.64
TOTAL PROPOSED COST	128.93	$ 1,164.33

SAVINGS

MONTHLY	$ 444.15
ANNUALLY	$ 5,329.80
PERCENTAGE	27.6%

800 SERVICE

CURRENT CONFIGURATION

SERVICE	HOURS	COST
(4) U.S. West Access Line Charges		$ 160.00
U.S. West Usage (1 minute rounding)	35.6	$ 409.40
AT&T Usage (30 second rounding)	54.0	$ 718.20
TOTAL COST FOR 800-228-0978	89.6	$ 1,287.60
(2) U.S. West Access Line Charge		$ 80.00
U.S. West Usage (1 minute rounding)	6.7	$ 77.06
AT&T Usage (30 second rounding)	15.5	$ 206.16
TOTAL COST FOR 800-541-8981	22.2	$ 363.22
TOTAL COMBINED COST	111.8	$ 1,650.82

PROPOSED CONFIGURATION

T-1 ACCESS

SERVICE	HOURS	COST
(6) MCI 800 Access Lines		$ 0.00
MCI 800 Usage (6 second rounding)	111.8	$ 915.13
TOTAL PROPOSED COST	111.8	$ 915.13

SAVINGS

MONTHLY	$ 735.69
ANNUALLY	$ 8,828.28
PERCENTAGE	44.6%

COMBINED SAVINGS

T-1 ACCESS

MONTHLY	$ 1,179.84
ANNUALLY	$ 14,158.08
PERCENTAGE	36.2%

The cost for T-1 Access would be $256.46 per month and $1,184.08 to install. However, we currently have a T-1 promotion which allows customers who subscribe between February 1, 1989 and March 31, 1989 and agree to installation no later than June 30, 1989 one month free access and free installation. This represents a cost savings of $1,440.54.

Long Distance Cost Summary

Service		January	February	March	April	May	June	July	August	September	October	November	December	Average	Action level
ODD	MIN	2365	3456	2321	2346	3112	3432	1331	3219	3234	1321	2128	4443	2809	4000
	COST	$683.49	$953.86	$652.20	$642.60	$774.69	$912.91	$652.68	$904.54	$899.05	$365.35	$597.07	$1,236.15	$772.07	$1,200.00
	COST/MI	$0.29	$0.28	$0.28	$0.27	$0.25	$0.27	$0.26	$0.28	$0.28	$0.27	$0.28	$0.28	$0.28	$0.29
BAND 1 WATS	MIN	4002	4523	5423	4322	5423	4327	5674	2345	5433	2367	4675	3432	4329	6000
	COST	$956.48	$1,085.52	$1,290.67	$972.45	$1,279.83	$1,064.44	$1,299.35	$579.22	$1,266.89	$591.75	$1,140.70	$854.57	$1,031.74	$1,500.00
	COST/MI	$0.24	$0.24	$0.24	$0.23	$0.24	$0.25	$0.23	$0.25	$0.23	$0.25	$0.24	$0.25	$0.24	$0.25
BAND 5 WATS	MIN	4922	5563	6670	5316	6670	5322	6979	2884	6683	2911	5750	4221	5324	7000
	COST	$1,426.23	$1,618.65	$1,924.56	$1,450.05	$1,908.39	$1,567.22	$1,937.49	$863.68	$1,887.60	$862.38	$1,700.93	$1,274.27	$1,538.48	$2,100.00
	COST/MI	$0.29	$0.29	$0.29	$0.27	$0.29	$0.30	$0.28	$0.30	$0.28	$0.30	$0.30	$0.30	$0.29	$0.30

TITLE: Implementing Micros in an HP 3000 Environment

AUTHOR: Birket Foster (613) 230-4321
M.B. Foster Associates Ltd.
P.O. Box 580
Chesterville, Ontario K0C 1H0 Canada

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 4006

EXPLOITING THE POWER OF YOUR HP TERMINAL EMULATOR

By Ernesto J. Morales
and
John R. Moffitt

Compaq Computer Corporation
P.O. Box 692000
M410302
Houston, TX 77269-2000

(713) 374-2932

INTRODUCTION

This paper will illustrate several techniques to take advantage of the power of terminal emulation software in a manner easily understandable by users and programmers.

Although the examples in this paper refer specifically to Reflection, an emulator distributed by Walker Richer & Quinn, the means of communicating with most emulators, and even with dumb terminals and printers, are usually the same. Therefore, many of the techniques mentioned in this paper can be useful even to users that do not have Personal Computers available to them or are not running Reflection but some other emulator.

Personal Computers are becoming ubiquitous and substituting dumb terminals as a means to interact with an HP computer. This is accomplished by running terminal emulation software. Some of the emulation packages are very powerful and allow the user to alter the terminal characteristics and interact with the PC operating system. However, most of their features go unexploited because few users are technical enough to understand them and many programmers do not read the manuals for lack of time or interest.

Generally, a user or a program communicates with the emulator with escape sequences. An escape sequence is a string which first character is ESC (decimal 27). Some of the programmers that do use the these sequences in their programs include the ESC character in their code. This causes problems when listing the programs, since the escape sequences are interpreted and obeyed by the emulator, rather than displayed normally.

Even when the escape character is defined in a different manner, escape sequences are obscure. CHAR(27) // "&p7p20C", for example, does not mean much to a programmer trying to modify a program.

In this paper we present several subroutines, UDCs (User Defined Commands), programs and Reflection command files that we have developed to take full advantage of Reflection and write easily understandable code. These techniques simplify both the user's and the analysts' job.

FEATURES OF THE HP TERMINAL EMULATOR

An emulator is a program that allows your Personal Computer to communicate with a larger computer system (called the "host" computer) as if you were using a terminal. Reflection emulates, among others, HP3000 terminals. As with a terminal, you can change the configuration. For example, you can turn LOG BOTTOM or RECORD MODE on and off, reset the terminal, move the cursor, set tabs and margins, define the user keys, lock the keyboard, set local echo on and off, begin and end enhancements, roll the display up or down, transfer data from display memory to the printer, etc.

Additionally, Reflection allows you to:

* Have a much larger display memory.
* Transfer data from display memory to the PC.
* Transfer files between the HP and the PC.
* Create, rename, and erase files on the PC.
* Display data on the screen as if it had been sent by the HP.
* Use configuration files with different set-ups. For example, one to logon using a modem, and another one to logon with a direct connect.
* Send strings to the HP, that can be interpreted as commands or as answers to prompts, and wait for a specific host response or a period of time.
* Backup your PC files to a file on the HP.
* Run any PC Operating System command or program. You could, for example run Lotus or even reformat your hard drive from the emulator.
* Run the emulator on the background, while you run other software on the PC.
* Write programs using the command mode for quick execution of commonly performed operations.

All these features, and more, can be used based on all sort of conditions. The emulators allow you to define variables, use functions to alter them, determine error conditions, prompt the user, and more.

A program running on the HP can communicate with the emulator the same way that it would communicate with a dumb HP terminal: by sending Escape Sequences. An escape sequence is a string that starts with an ESC character (decimal 27). For example, if the emulator receives the escape sequence

ESC h

instructs the emulator to home the cursor.

SUBROUTINES

As mentioned earlier, there are a few problems that can be encountered when coding programs that take advantages of the emulator's features by sending escape sequences. If the ESC character is hard-coded in the program, listing the program becomes a problem, since the escape sequence will be obeyed by the emulator. If, for example, an instruction in the program blanks the screen with a escape sequence, the screen will go blank when you list the line of code containing that instruction.

Also, escape sequences are not self-explanatory at all. In general they are very obscure. For example, the escape sequence to turn log bottom on is ESC concatenated with "&p0p4u11C". Most probably, if a programmer maintaining the code encounters this sequence he will have a hard time understanding what it does.

We have developed a set of subroutines that take care of the problems with coding escape sequences in programs. Their use results in very clear and easily maintainable programs. In a FORTRAN program, for example,

```
PRINT *,CHAR(27),"&p0p4u11C"
```

is substituted by

```
CALL LOG_BOTTOM_ON.
```

The subroutines were written as entry points to a single subroutine in FORTRAN. They are very simple. These and many other similar routines can be easily written in any other language.

A couple of entry points illustrate the power of the emulator and the benefits that the programmer and the user derive from using this technique: SET_VPLUS_CONFIG and LOAD_USER_CONFIG. All the programmer needs to do is include a

```
CALL SET_VPLUS_CONFIG
```

statement at the beginning of the program that uses VPLUS, and a

```
CALL LOAD_USER_CONFIG
```

at the end of the same program. When the first is executed, it saves the user configuration to a file in the PC, and then sets the configuration to run a VPLUS program in such a way that the user can use the "enter" key to send data to the program, eliminating the awkward use of the "enter" key for one type of input (character mode programs, QUICK) and the "plus" key for another (VPLUS screens). When the second routine is executed it restores the user's configuration to the way it was before the VPLUS program ran, including his function key definition.

The other entry points that we have developed are:

```
DISABLE_COMP_CODES
ENABLE_COMP_CODES
DO_FORM_FEEDS
NOT_DO_FORM_FEEDS
SAVE_USER_CONFIG
RETURN_EQ_ENTER
RETURN_NE_ENTER
INHIBIT_EOL_WRAP
ENABLE_EOL_WRAP
RECORD_MODE_ON
RECORD_MODE_OFF
DISPLAY_ON
DISPLAY_OFF
MEMORY_LOCK
MEMORY_UNLOCK
LOG_BOTTOM_ON
LOG_BOTTOM_OFF
COMPRESS_PRINT
NORMAL_PRINT
SET_RIGHT_MARGIN (margin)
SET_REPORT_CONFIG
OPEN_PC_DISK  (pc_name)
CLOSE_PC_DISK
OPEN_PC_PRINT
CLOSE_PC_PRINT
```

The following is a subset of the source file where these subroutines were coded:

```
      SUBROUTINE REFLECTION
      IMPLICIT NONE
      SYSTEM INTRINSIC ASCII
      INTEGER COLUMN,DUMMY,FIN
      CHARACTER XCOLUMN*3, PC_NAME*30

C-----------------------------------------------------------------------
C     DISABLE COMPLETION CODES. NORMALLY, REFLECTION SENDS AN "S" OR
C     AN "F" INDICATING WHETHER A COMMAND WAS SUCCESSFUL OR NOT. THIS
C     SUBROUTINE DISABLES THAT FEATURE. THIS ENTRY POINT ALSO EXECUTES
C     WITH A CALL TO SUBROUTINE "REFLECTION".
C-----------------------------------------------------------------------
      ENTRY DISABLE_COMP_CODES
      PRINT *,CHAR(33B),"&oC","SET DISABLE-COMP-CODES YES",CHAR(15B)
      GOTO 900

C-----------------------------------------------------------------------
C     SAVES THE CURRENT REFLECTION CONFIGURATION TO A DISK FILE IN THE
C     CURRENT DRIVE AND DIRECTORY (ON THE USER'S PC).
C-----------------------------------------------------------------------
      ENTRY SAVE_USER_CONFIG
      PRINT *,CHAR(33B),"&oC","SAVE DELETEME.XYZ DELETE",CHAR(15B)
      GOTO 900
```

```
C-------------------------------------------------------------------
C     CAUSES THE RETURN KEY (LABELED "ENTER" ON MOST PC KEYBOARDS) TO
C     PERFORM AS AN HP TERMINAL ENTER KEY ("+" KEY ON THE NUMERIC
C     KEYPAD OF MOST PC s). VERY USEFUL FOR VPLUS PROGRAMS.
C-------------------------------------------------------------------
      ENTRY RETURN_EQ_ENTER
      PRINT *,CHAR(33B),"&oC","SET RETURN=ENTER YES",CHAR(15B)
      GOTO 900

C-------------------------------------------------------------------
C     ASSIGNS <COLUMN> COLUMNS AS THE PAGE WIDTH, WHERE <COLUMN> IS A
C     NUMBER IN THE RANGE 1-999. IT HAS THE SAME EFFECT AS MANUALY
C     CHANGING THE "RIGHT MARGIN" FIELD IN THE "TERMINAL PAGE 1"
C     CONFIGURATION SCREEN.
C-------------------------------------------------------------------
      ENTRY SET_RIGHT_MARGIN(COLUMN)
      DUMMY = ASCII (COLUMN,10,XCOLUMN)
      FIN = 3
      IF (COLUMN.LT.100) FIN = 2
      IF (COLUMN.LT.10)  FIN = 1
      PRINT *,CHAR(33B),"&q3t0(",XCOLUMN(1:FIN),"W"
      GOTO 900

C-------------------------------------------------------------------
C     THIS SUBROUTINE REDUCES THE NUMBER OF CALLS NEEDED IN  A VPLUS
C     PROGRAM TO SET THE TERMINAL CONFIGURATION. SEE THE INDIVIDUAL
C     SUBROUTINE DESCRIPTIONS ABOVE FOR DETAILED INFORMATION.
C-------------------------------------------------------------------
      ENTRY SET_VPLUS_CONFIG
      CALL DISABLE_COMP_CODES
      CALL SAVE_USER_CONFIG
      CALL SET_RIGHT_MARGIN(80)
      CALL RETURN_EQ_ENTER
      GOTO 900

C-------------------------------------------------------------------
C    THIS SUBROUTINE WILL OPEN THE "to" DEVICE AS A PC DISK FILE
C    WITH A USER SUPPLIED FILE NAME.  THIS ROUTINE 'ASSUMES' THAT
C    NO PRINTER OUTPUT IS DESIRED & TURNS OFF THE "TO PRINTER".
C    IF SIMULTANEOUS PRINTER OUTPUT IS DESIRED, CALL OPEN_PC_PRINT
C    AFTER CALLING OPEN_PC_DISK.
C-------------------------------------------------------------------
      ENTRY OPEN_PC_DISK (PC_NAME)
      IF (PC_NAME.EQ." ") PC_NAME = "NO_NAME.DUM"
      CALL CLOSE_PC_PRINT
      PRINT *, CHAR (33B), "&oC", "OPEN ", PC_NAME, " DELETE",
     *         CHAR (15B)
      GOTO 900

900   RETURN
```

The following code shows how a FORTRAN program creating QUIZ as a son process can send an output report to a PC file:

```
.
.
IF (PCDISK) THEN
    INFO_PARM = "CC=(PCDISK) NOLIST"
    CALL DISABLE_COMP_CODES
    CALL OPEN_PC_DISK (PCFILE)
    CALL LOG_BOTTOM_ON
    CALL COGNOS_CREATE ("QUI2", QUIZ_NAME, STDIN, STDLIST,
*                       INFO_PARM,0,RET)
    CALL LOG_BOTTOM_OFF
    CALL CLOSE_PC_DISK
    CALL ENABLE_COMP_CODES
ELSE
.
.
```

The QUIZ program could, optionally, not print page headings and footings when the output is sent to a PC file, and print them when it is sent to a printer, as follows:

```
@IF NOT PCDISK
        PAGE HEADING        &
        .                   &
        .
@ENDIF
.
.
@IF NOT PCDISK
        FINAL FOOTING       &
        .                   &
        .
@ENDIF
.
.
GO
EXIT
```

UDCs

The following UDCs demonstrate how simple it becomes for the user to take advantage of the features of the emulator:

```
**********
LPI8
COMMENT
COMMENT    SETS LINES PER INCH TO 8 (PRINTER)
COMMENT
SETVAR ESC CHR(27)
SETVAR REC_MODE_ON  ESC + "&p7p20C"
SETVAR REC_MODE_OFF CHR(7)
SETVAR EIGHT_LPI    ESC + "0"
ECHO !REC_MODE_ON
ECHO !EIGHT_LPI
ECHO !REC_MODE_OFF
DELETEVAR ESC
DELETEVAR REC_MODE_ON
DELETEVAR REC_MODE_OFF
DELETEVAR EIGHT_LPI
**********
LPI6
COMMENT
COMMENT    SETS LINES PER INCH TO 6 (PRINTER)
COMMENT
SETVAR ESC CHR(27)
SETVAR REC_MODE_ON  ESC + "&p7p20C"
SETVAR REC_MODE_OFF CHR(7)
SETVAR SIX_LPI    ESC + "2"
ECHO !REC_MODE_ON
ECHO !SIX_LPI
ECHO !REC_MODE_OFF
DELETEVAR ESC
DELETEVAR REC_MODE_ON
DELETEVAR REC_MODE_OFF
DELETEVAR SIX_LPI
**********
LOADUSERCONFIG
COMMENT
COMMENT    LOADS A CONFIGURATION FILE SAVE PREVIOUSLY
COMMENT    TO THE PC FILE DELETEME.XZY
COMMENT
SETVAR W_LOAD CHR(27) + "&oCLOAD DELETEME.XYZ"
ECHO !W_LOAD
DELETEVAR W_LOAD
**********
```

```
SETVPLUSCONFIG LABEL=2
COMMENT
COMMENT    SAVES THE CURRENT CONFIGURATION TO A FILE ON
COMMENT    ON THE PC, AND SETS VPLUS CONFIGURATION,
COMMENT    INCLUDING "RETURN=ENTER"
COMMENT
SETVAR W_CMD CHR(27) + "&oCSET DISABLE-COMP-CODES YES"
ECHO !W_CMD
SETVAR W_CMD CHR(27) + "&oCSAVE DELETEME.XYZ DELETE"
ECHO !W_CMD
SETVAR W_CMD CHR(27) + "&oCSET RETURN=ENTER YES"
ECHO !W_CMD
SETVAR W_CMD CHR(27) + "&oCSET LABEL-LINES !LABEL"
ECHO !W_CMD
DELETEVAR W_CMD
SETVAR W_RIGHT_MARGIN CHR(27) + "&q3t0{80W"
ECHO !W_RIGHT_MARGIN
DELETEVAR W_RIGHT_MARGIN
**********
FORMS FORMFILE=BOX86
COMMENT
COMMENT    RUNS FORMSPEC WITH "RETURN=ENTER"
COMMENT
FILE FORMLIST=!FORMFILE;DEV=LASER
SETVPLUSCONFIG
RUN FORMSPEC.PUB.SYS
LOADUSERCONFIG
**********
```

LPI8 and LPI6 can be used by the user to vary the number of lines per inch in his printer. FORMS uses LOADUSERCONFIG and SETVPLUSCONFIG to save the user's configuration, run FORMSPEC (a VPLUS program) allowing the user to use the "enter" key instead of the "plus" key, and finally restore the user configuration, including his function key definition.

Many more UDCs can be created. For example, files can be moved between the PC and the HP.

PROGRAMS

The UDCs mentioned in the previous section would only work with MPE-XL, since some of the commands used do not exist in MPE V. However the same results can be obtained by running a program such as the following one:

```
$CONTROL STANDARD_LEVEL SYSTEM, SHORT
C*****************************************************************
C PROGRAM     : CONFIG
C PROGRAMMER : ERNESTO J. MORALES
C DATE        : 02/26/88
C DESCRIPTION: THIS PROGRAM EXECUTES A DIFFERENT REFLECTION COMMAND
C              WHEN RUN AS
C                 :RUN CONFIG;PARM=n
C              WHERE n CAN BE ONE OF THE FOLLOWING:
C                 1 = SET VPLUS CONFIGURATION
C                 2 = SET VPLUS CONFIGURATION + SET LABEL-LINES 1
C                 3 = SET REPORT CONFIGURATION
C                 4 = SET REPORT CONFIGURATION + RECORD MODE
C                 5 = SET REPORT CONFIGURATION + PRINT
C                 6 = LOAD USER CONFIGURATION + RECORD MOD OFF
C                 7 = LOAD USER CONFIGURATION + PRINT OFF
C                 8 = LOAD USER CONFIGURATION
C                 9 = LOAD \REF\DIRECT
C              IT CAN ALSO PERFORM ANY ARBIRARY ESCAPE SEQUENCE BY
C              SPECIFYING AN INFO PARAMETER
C*****************************************************************
      PROGRAM CONFIG(PARM,INFO)
      INTEGER*2 PARM
      CHARACTER*(*) INFO
      IF (PARM.GE.1 .AND. PARM.LE.5) THEN
          CALL SAVE_USER_CONFIG
          IF (PARM.EQ.1 .OR. PARM.EQ.2) THEN
              PRINT *, "Setting VPLUS configuration"
              CALL SET_VPLUS_CONFIG
              IF (PARM.EQ.2)PRINT *,CHAR(33B),"&oCSET LABEL-LINES 1",
     *                               CHAR(15B)
          ELSE
              PRINT *, "Setting wide report configuration"
              CALL SET_REPORT_CONFIG
              IF (PARM.EQ.4) THEN
                  CALL COMPRESS_PRINT
                  CALL RECORD_MODE_ON
              ELSEIF (PARM.EQ.5) THEN
                  CALL COMPRESS_PRINT
                  CALL LOG_BOTTOM_ON
              ENDIF
          ENDIF
      ELSEIF (PARM.GE.6 .AND. PARM.LE.8) THEN
          PRINT *, "Restoring user configuration"
          IF (PARM.EQ.6) THEN
              CALL RECORD_MODE_OFF
              CALL NORMAL_PRINT
          ELSEIF (PARM.EQ.7) THEN
              CALL LOG_BOTTOM_OFF
              CALL NORMAL_PRINT
          ENDIF
          CALL LOAD_USER_CONFIG
      ELSEIF (PARM.EQ.9) THEN
          PRINT *,CHAR(33B),"&oCLOAD \REF\DIRECT",CHAR(15B)
          PRINT *,"Configuration restored"
      ENDIF
      IF (INFO.GT." ") PRINT *,CHAR(33B),INFO,CHAR(15B)
      END
```

With this program the following FORMS UDC would do on an MPE V machine what the FORMS UDC explained in the previous section accomplishes on an XL machine:

```
FORMS
RUN CONFIG;PARM=1
RUN FORMSPEC.PUB.SYS
RUN CONFIG;PARM=8
```

The above program can also be used to send any escape sequence to the emulator. For example, the escape sequence to turn display functions mode on is ESC + "Y". If the program were executed as follows, it would turn this feature on:

```
:RUN CONFIG;INFO="Y"
```

EMULATOR COMMAND FILES

Reflection allows you to create command files. These are files that contain a series of Reflection commands. They are equivalent to programs. Many different applications can be given to command files. Below we show a command file that we use to get phone messages stored in a computer. This computer is not an HP3000. Reflection, and many other emulators, can be used to communicate with many host computers.

```
BREAK
WAIT 0:0:1
TRANSMIT "^M"
WAIT FOR 0:0:1 SILENCE
TRANSMIT "THE-COMPUTER-NAME^M"
WAIT 0:0:4
TRANSMIT "^M"
WAIT 0:0:5 FOR "login:"
IF NOT FOUND
GOTO OOPS
ENDIF
WAIT 0:0:1
TRANSMIT "MY-LOGIN^M"
WAIT 0:0:20 FOR "extension:"
IF NOT FOUND
GOTO OOPS
ENDIF
WAIT 0:0:1
TRANSMIT "MY-EXTENSION^M"
WAIT 0:0:20 FOR "password:"
IF NOT FOUND
GOTO OOPS
ENDIF
WAIT 0:0:4
TRANSMIT "MY-PASSWORD^M"
WAIT 0:0:5 FOR "COMMAND:"
IF NOT FOUND
GOTO OOPS
ENDIF
OPEN PRINTER
LOG
WAIT 0:0:1
TRANSMIT "LI U^M"
WAIT 0:1:0 FOR "type of messages:"
IF NOT FOUND
GOTO ALMOST
ENDIF
:ALMOST
LOG OFF
PRINT "^L"
TRANSMIT "Q^M"
:OOPS
STOP
```

Command files in Reflection can be executed when the emulator is started, by typing ALT-Y to invoke them, or with a escape sequence. The following BAT file runs Reflection to get phone messages automatically:

```
@echo off
CD \REF
R1 DIRECT,MCS.CMD
```

CONCLUSIONS

The techniques explained in this paper are only examples of what can be done to take full advantage of the terminal emulators. Most of them are very simple and can be set up with little effort.

We have documented these and many more techniques and included them in our programming standards. This has paid large dividends in the form of:

1. Reduction of program development time.
2. Faster turnaround for program modification.
3. High quality computer code that is both easy to understand and maintain.
4. Simplicity and efficiency for the end user.

We encourage you to not only to use the above techniques, but expand on them[1] and create as many subroutines, UDCs, programs, command files, and combinations of these as you and your users need.

---

[1] J. Moffitt and E. Morales, Synergistic Programming, Interact, July 1989, pp. 50-60

Is the PAIN Worth the GAIN?!?!

Asher F. Tunik
Computer Task Group
One Commerce Center
12th and Orange Streets. Suite 1000
Wilmington, DE 19801

So, your organization decided to implement a system that uses portable computers? This paper will discuss several issues that are involved in a laptop project. We will start with the Implementation issues, then continue with Training, and finally spend some time on System Support. After these issues are covered, we will talk about the path forward and new possibilities that could open with the use of laptops.

Implementation Issues

The four major issues that fall under the category of implementation are:

- Business Support
- System Design
- Development
- Roll-out

The most important implementation issue of any project is **BUSINESS SUPPORT**. The project must have a strong sponsorship from both the user community and their management. If only one group (user community or management) supports the project it is sure to fail. If only management is going to use the output from the system, users will find a way to fool the system or not use the system at all. If it is only the users who want to use the system, chances are management will find a way to direct the energy of their subordinates to pursue some other goals. So, before even starting the design phase, talk to the user community, talk to their management, and ask them why they need the system (the answers may surprise you). Also ask if there are other ways of doing whatever it is they are doing, and should there be such an investment as implementation of a new system?

One way to make a system wanted is to provide a tool that will help the users do their job better and/or faster. A good example might be providing 123 macros that do hours worth of manual calculation in return for the effort of putting in the numbers, or the ability to provide prospects with professional quality graphs and charts that would not even be possible before the use of computers. Another way to help the users might be to give them the ability to submit their expense reports electronically, thus reducing the time it takes to get their expense money back.

Since most laptop systems are used primarily by the user community, the value provided to the users should be of primary concern. Whatever value management receives should be secondary. This concept sometimes makes laptop projects difficult to justify financially. Management, however, can gain value from the central computer system that communicates with the laptop system. (We will talk about such a system and communication mechanisms a little later.) Some other justifications can be intangible: improved morale, better and easier communication, rapid receipt of data, etc.

SYSTEM DESIGN is dictated by the software functionality. Before starting the design phase, make sure that all grounds are covered and you have a good understanding as to what the system should do in order to satisfy user requirements and to fit into the "BIG" MIS picture. In addition to the standard computerized system design issues, there is a need to be able to easily exchange versions of the software and data updates. If possible, design an electronic inventory system. This is a way for the host to keep track of exactly what each laptop should have on it. It should include program versions and actual data. This is the only sure way to back up the laptop. It also reduces the administrative effort needed to distribute new versions of the software.

Let us talk a little bit about different ways to **DEVELOP** the system. As with all the systems, there are two general approaches: Prototyping and Structured development.

When prototyping is chosen, it is critical to select a representative pilot group. This group needs to be small enough to provide personal attention to, and large enough to represent a true cross section of the user community

(outstanding, good, average and bad). If possible also include a cross section of user management.

A feeling of ownership is very important; let the pilot group design the screens and report layouts. If possible, put as much of the functionality that the group requests into the prototype. (If it is not needed, it can be removed from the real system later.) If your data dictionary allows it, use variable names that are meaningful to the group. One potential problem with the prototyping is knowing when to stop. As the pilot group learns more and more about what the system can do, they will think up more functionality for it. To some degree, this is exactly what is desired, but after a point it can lead to a far larger system than is needed and/or can be afforded. Another problem is selling the prototyping approach to old line structured development Data Processing management. If this is a problem, do it the structured way.

If the project scope is known fairly well up front, the structured approach can be appropriate. The structured approach, however, takes more time, which can lead to a loss of user interest in the development effort, which may spell certain failure.

Let us talk a little bit about methodology of such a project. A centralized database will be needed to work as a central repository for data. Having a centralized database will allow easier interface with other major systems. Every user will have only a portion of the data that he/she uses. On the laptop, each user can change data that he/she "owns", which can also be changed in the central location (for example, in a corporate or regional office). At some established time there will be a connection between the laptop and the central computer, at which time data will be synchronized. Users of the laptops will get their primary value out of using data on laptops. Management can have reports coming out of the central database and make decisions based on these data.

When you finally have a system ready, how do you go about the ROLL-OUT to the users?! I am sure that there are many different ways to implement the roll-out. Two distinct approaches are: All-at-Once (everybody across the board) and Phased (usually by district, territory or region).

The All-at-Once approach can lead to simpler logistics. You have more laptops to deal with, but you only need to do the set up, testing, distribution, etc., once. You can put on a "nice show" and try to generate a lot of enthusiasm. Since this approach costs somewhat more money, planning is critical. There are a lot of things that can go wrong (as with any theatrical production), so plan for everything going wrong that you can think of.

With the All-at-Once approach, it is a good idea to provide follow-up training in smaller groups. Some preliminary training should be given when the machines are handed out (how to turn them on, how to load the software, and how not to break the hardware).

The other roll-out approach is Phased implementation by some organizational or territorial grouping. This approach seems simpler, but since the roll-out needs to be reproduced several times, logistics can become complicated. An example of a problem could be that not all the sites can provide enough outlets to charge the devices.

Some of the advantages to a phased roll-out are that more detailed training can be provided at the time of the roll-out, and you can also learn from your mistakes; whereas in the All-at-Once approach if you made a mistake all the users will see it.

TRAINING ISSUES

There are four major issues that fall under the category of training:

- o Training users how to use the software
- o Training users how to use and take care of the hardware
- o On-going training for the veterans
- o Training for the new users and support staff

TRAINING USERS HOW TO USE THE SOFTWARE is the most important part of the training; it cannot be overdone. An obstacle in this area is that a busy professional usually can devote only a limited contiguous time to a training session and, therefore, training in frequent short sessions will be most acceptable.

Another thing to remember when developing a training program is that laptop users use computers to help them do their jobs better. That is what the bulk of the training should focus on. DOS internals are fascinating, however users probably should not even see the DOS prompt; keep technology training to an absolute minimum.

Provide users with a lot of hands on practice, which in turn means lots of individual attention. Keep the groups small or have enough instructors at the session, so that a lot of patient, individual attention can be given to each trainee.

Since people often ask their boss to help them solve problems, it is important that their boss knows the system. It is very important that first line supervision understands the new system, knows how to use it, and knows what it can and cannot do.

TRAINING USERS HOW TO USE AND TAKE CARE OF THE HARDWARE is also a very important aspect of the training. This type of training should happen fairly close to the time that people get their laptops. Show people how to change the battery, how to change a ribbon or ink cartridge in the printer, what kind of replacement battery and ribbon they need to buy, etc. Users probably do not need to know how to replace the mother-board, but they should know

how to handle a few simple hardware problems. A good idea would be to create a handout book with "HOW TO" pages. Such a book should be in the form of a three-ring binder, so additional pages can be inserted easily.

Organizations have personnel turnover. If the system is successful it will be around for a long time. That is why there is a need for **ON-GOING TRAINING FOR THE VETERANS** and **TRAINING FOR THE NEW USERS**. New users need complete training, not just the continuing training that experienced users are getting.

SYSTEM SUPPORT ISSUES

SUPPORT!!! Most Data Processing professionals dread this word. While support is expensive, boring, time consuming, and irritating, it is the MOST CRITICAL part of the laptop based system.

Quite often when users call for help, they are at the customer site, and it is impossible to call them back. Often they are irritated and in a hurry. That is why an immediate answer is an important issue with the users of the laptop based system. All of the above reasons dictate special constraints on the people in support. They should be patient, understanding, able to establish a good rapport with the users, and on top of all that they must be KNOWLEDGEABLE; not only in software and hardware but also knowledgeable of how your organization does business and interacts with customers.

There are three major issues that fall under the category of the system support:

- o Software Support
- o Procedural Support
- o Hardware Support

SOFTWARE SUPPORT is a rather obvious issue. Someone who is in system support should know how the software works and what it can and cannot do. **PROCEDURAL SUPPORT** is not as obvious. Sometimes users call in with a question that goes something like: "Hey, what do I do when....?" Support personnel should be able to at least direct the question to the right person, or maybe even answer it. When Software Support and Procedural Support questions arise, it is a good idea to create a "How To" page and send it out to all the users, so they can insert it into their three-ring binder (mentioned earlier).

If the project is successful, users will utilize their equipment, they will grow to rely on it to do their job, and quite often they will not be able to do their job without it. Because of this, **HARDWARE SUPPORT** is a critical issue. Replacement systems should be sent to the user via overnight courier. System Support personnel should have an inventory of the replacement components (printers, disc drives, laptops, etc.) and when an inoperable machine is

received, it can be fixed and put back into the inventory of replacement components.

Path Forward and Additional Capabilities

Once the system is in place and operational and all the start-up jitters are safely behind, it is time to see how the use of laptops can be extended with minimal expenditure of effort.

An obvious example would be use of electronic mail. Laptop users can logon to the central computer and exchange electronic mail with their colleagues and other members of the user community. No more "telephone tag", communication is better and faster, and it is achieved without any extra expenditures of capital or effort.

Another example can be the use of standardized forms. A blank form in a wordprocessor format can be distributed to all users. For example, when users need to fill out a form, instead of doing it on paper they can fill in the electronic form and transmit it to the central office electronically.

With Cellular communications becoming more and more popular, it is inevitable that soon there will be reliable and affordable cellular modems on the market. Can you think of any additional capabilities by using an on-line communication with a central computer without utilizing a phone line? Service organization can generate payable invoices at the time a service is performed, instead of the 45-60 days needed to:

- Send a Field Service Report to the Central Office,
- Wait for the Central Office to process the Field Service Report and send a request for an invoice to Accounts receivable,
- Wait for Accounts Receivable to generate the Invoice,
- Mail the Invoice to the customer

The point is: Additional uses for laptops and additional functionality is limited only to the extent of your imagination.

Conclusion

There are a lot of similarities between conventional computerized systems and systems that utilize laptops. However systems that utilize laptops have certain unique attributes that must be addressed in order for the project to be successful.

Bibliography:

1. Treisner, George, "Laptops at Du Pont Medical Products", Presentation given at NECRUG conference May, 1989.

2. Tunik, Asher F., "Field Service Automation: Benefits and Pitfalls", Proceedings from INTEREX HP Users Conference, September 11-14, 1989 (paper number 3900)

3. Levine, Ron, "Take Good Care Of Your Desktop", HP Professional, August, 1989, pp 98-99.

The Role of the Consumer in Strategic Planning

Lloyd D. Davis
School of Education
The University of Tennessee at Chattanooga

Elisabeth M. Craig
Center of Excellence for Computer Applications
The University of Tennessee at Chattanooga

Abstract

During the summer of 1988 a study team at the University of Tennessee at Chattanooga (UTC) prepared a strategic plan for academic computing for the years 1988-92. The seven member Strategic Planning Study Team, was composed of representatives from each of the colleges and schools (one member was from Academic Computing). The team initially identified 12 major categories (such as access to facilities, security, etc.) and discussed the issues involved in each of these categories. Then, the study team prepared a 14-question survey which was distributed to 70 faculty and administrators from all disciplines. The questions were designed to analyze reactions to categories of support services, to examine perceptions about specific issues and to construct a priority of needs. These individuals represented the consumers of the services provided by Academic Computing. A week later, 67 of these respondents were interviewed for an hour in groups of four or five. The written surveys and interviews provided extensive qualitative data about long-range needs of the respondents. Using the constant comparative method devised by Glaser and Strauss, the data was transformed into a matrix of needs and priorities. In turn this matrix was used to develop recommendations submitted to the administration. In addition, timetables and schedules for the estimated costs of the strategic plan were created.

The paper describes this procedure for strategic planning and provides an overview information about the data collected. Another topic which is explored are the strengths and weaknesses of consumer input to the strategic planning process and how to relate the results to the mission of the institution.

The paper also discusses the fact that the process is repeatable, has a strong focus and examines how well the method identified priority issues on a qualitative basis.

The participatory nature of this method of strategic planning assures that the results are viable and that they are accepted as such by the administration and by the consumer. While the consumers at UTC were not business managers *per se* there is no reason to suppose that most of the procedures could not be effectively used in other environments.

Introduction

Strategic planning most often flows from the mission statement of an institution which states the specific goals and objectives defined by top management. This is often called a business plan. In the summer of 1988, The University of Tennessee at Chattanooga evolved and wrote a strategic plan that utilized the top-down source of direction, but also examined the issues within academic computing which were of vital interest to the most important constituency within the university, namely the faculty. Hence, the focus was on the user of computing - the consumer who was a faculty member representing his/her individual needs as well as those of the student.

This paper deals how to focus on discovering important issues among the consumers of a product. It also explores the quantification of the breadth and depth of these issues and discusses the construction of the strategic plan. A plan which had to be compatible with, and directly support, the institutional mission statement and top management goals and objectives. Such was the study done at The University of Tennessee at Chattanooga.

This paper's attention is directed primarily at the process of evolving and constructing this plan rather that at the institution specific recommendations which resulted from the procedure. Details are presented only to the extent that they help to convey information or illustrate the process.

Task Force Charge and Selection

A vendor approached the chief academic officer at the university and suggested creating a Task Force to produce a strategic plan for academic computing over a five year period. In return for access to the Task Force, the vendor agreed to contribute a full-time facilitator and two part-time assistants. Other vendors were contacted and invited to join in the process on the same basis, but declined to participate.

The seven members of the Task Force were selected by the administration. Five were from academic departments chosen so that each of the five colleges and schools were represented. The other two members were from the Library and from Academic Computing. One of the faculty members was chosen to be the Chairperson. All the members were computer knowledgeable and several of the team had extensive writing skills, a necessary factor since the final document contained over 100 pages.

Schedule of Process

The Task Force completed its work in six weeks as shown in the schedule below. There was one week between the first and second working weeks when the group did not met because the time was needed to allow respondents to complete a survey.

The Task Force met for six hours, four days a week, or in other words for 24 hours out of the normal 40 hour working week. Release time from normal duties was given to 12-month employees as it was impossible to perform a normal work load and spend the necessary hours on the Task Force. The same meeting room was use throughout to provide continuity and to establish a safe environment in which to store materials generated from the work. For example, large poster-size sheets of notes were taped to the walls as reminders and exhibits for the six week period.

Week 1	Topics for discussion were selected, and subtopics within each area were written down.
Week 2	Interviews were conducted.
Week 3	The main issues and problems were identified and categorized into a matrix.
Week 4	Members reviewed written statements and prepared recommendations based on the matrix.
Week 5	Recommendations were drafted addressing each of the areas identified in weeks 3 and 4. A timetable for implementation of the proposed solutions was developed.
Week 6	The implementation plan was drafted. The final report was edited and published.

Selection of Topics

In Week 1 the Team designated 12 major topic areas which were selected from a list provided by the vendor facilitator. Each topic was explored to determine the subtopics and also to gauge the importance of the items. An average of three hours was spent on each topic; the subtopics were noted and posted to large sheets of paper prominently displayed in the room. This procedure ensured that issues of importance to each member were discussed and listed.

The selected topics (in alphabetic not priority order) were:

Access
Communications?Networking
Computer Literacy
Consulting/Training
Faculty Rewards
Hardware
Instruction
Library
Policy and Organization
Public Service
Security
Software
Standards

In the final afternoon of the first week the Task Force split into two groups; each prepared 14 questions. These were written on transparencies and compared at a later time. Duplications were eliminated, some questions were merged and others eliminated to obtain the instrument. At the beginning of the week it seemed impossible that a survey of any worth could be constructed in an afternoon but the extensive debate during the week created a consensus which made the creation of the instrument flow smoothly. The final task of the week was to select the 68 respondents. The team invited all the academic deans and top administrators plus major computer consumers from the academic departments. Thus, each department, even those not using computer resources, were represented by at least one faculty member. In addition key staff (non-faculty) personnel were included as a mechanism to provide balanced input. Although students were not involved directly, their specific needs were articulated by the faculty.

Survey Instrument

The questionnaire sent to the 67 participants contained a total of 14 questions. The first one requested demographic information as to whether the respondent owned a microcomputer, which department he/she belonged to, the length of computer experience and the how the respondent used computers. The remaining questions were of the open-ended type and focused on major issues within the topics. Typical questions related to the role of new technologies in education, purchasing standards for hardware and software and perceived future needs. A copy of the survey appears in Table 1.

Interview Techniques

Those selected to participate represented all the academic disciplines and all administrative departments. Clerical support personnel phoned each interviewee to determine a convenient time. Of the 68 people who said they would attend, only one failed to do so. Almost without exception the interviewees arrived on time. The high attendance rate and punctuality indicated the responsiveness and issue identification which the interviewees ascribed to the process.

The 67 interviews were conducted during Week 2 (three people who were unavailable for interviews sent completed surveys for a total of 70 respondents). The format used was that three or four interviewees from the same discipline or administrative area met with all the members of the Task Force. Those interviewed had received the survey instrument at least a week earlier and were asked to bring ten completed copies with them. The interviews were based on responses to the survey questions though the respondents were encouraged to add comments or emphasis areas of special interest to the individual or the department.

Task Force members were instructed that their main purpose was to listen and to elicit information; comments and especially rebuttals were to be avoided. The idea

was that interviewees should feel that they had an attentive audience and that they could discuss the issues without conflict or stress.

Each interview period lasted for an hour and at the conclusion a half-hour was designated so that Task Force members could review notes and reach a consensus about the facts and opinions expressed by the interviewees. One Task Force member was designated to take official notes which were written on transparencies so that at the end of the interview all the Task Force could verify the notes. Members took turns in this note taking activity. At the end of each day clerical personnel transcribed the information written on the transparencies into computer format. At the end of the week Task Force members received a complete copy of all these transcribed notes.

Development of Matrix

With the interview process completed, the main issues were identified during weeks 3 and 4. The first task was to transform qualitative data (interview notes and completed surveys) from the 70 respondents into quantitative format for easier analysis. The constant comparative method, developed by Glasner and Strauss was used for this transformation. This method stresses the discovery of theory from a comparison of the data. The Task Force split into two teams; one examined data collected from interview groups 1 through 9 and the second group analyzed and synthesized information from the remaining nine groups. Respondents' responses were compared and placed into categories, based on the 12 major topics listed above. Each category and item within it was mutually exclusive from others in the matrix.

After the analysis and synthesis was completed a matrix of categories was derived (see Table 2). The matrix listed the major categories and the subcategories within each one. For the sake of brevity only the two top subcategories are listed in the table at the end of this paper. The matrix indicates how many individuals in each group addressed that specific issue, thus giving some idea of the impact. The column headed 'Pct Grps' indicates the percentage of groups (out of 19) which responded to the subtopic. The column marked 'Sum' shows the total number of respondents who discussed the subtopic, while under 'Pct Resp' is reported the percentage this represented. For example, under the category Consulting/Training 18 of the 19 groups (or 95%) discussed hardware/software support; also 46 out of the 70 respondents (or 65%) made comments about this issue.

Five Year Strategic Plan

The matrix which became the basis for the strategic plan and for recommendations developed in weeks 4 and 5. For example, the top two rated issues under Consulting/Training were hardware/software support and short courses/seminars which resulted in the recommendation for enlarged user services in the strategic plan to address these needs. The recommendations were reviewed

for compatibility with the university's mission plan and were included when found to be supportive. For example, the recommendations for Consulting/Training were:

- Enhance and enlarge User Services.
- Establish a hot-line, staffed by a skilled person, to screen calls and direct the caller to the right trouble-shooter.
- Review, acquire, and publicize appropriate microcomputer software.
- Prepare a selected list of hardware and software that will be supported by Academic Computing.
- Make documentation for computer hardware/software more widely available.
- Create an extensive faculty training program on the use of computers in the classroom
- Offer courses to enable students to become computer literate.

Also, the matrix became the framework for a five year cost analysis which listed all proposed projects. This plan showed the dollar amounts needed for fiscal years 1988-89 through 1992-93 as well as a status (e.g. initiate, continue, update etc.) for that project for each of those years.

The strategic plan, plus supporting documentation, were presented to the administration and later were circulated to all 67 respondents.

This paper has focused on the actual process of using a Task Force rather than on its recommendations. The rationale for this approach is that organizations wanting to use this consumer-oriented method need information on the overall procedures rather than on the recommendations and cost analysis which are specific to The University of Tennessee at Chattanooga.

Conclusions

In addition to the development of the matrix and five year plans there were other intangible by-products of the Task Force process. First, the fact that the process was completed within a seven week period created an intensive environment and a consensus opinion among the team not obtainable by a longer term process. Second, all members of the Task Force received valuable training in the practical aspects of strategic planning as well as an overall knowledge of computing activities on the campus. Third, the process involved all sectors of the institution and allowed users to ventilate about past procedures and to indicate their specific needs as well as those of their departments. User priorities and future needs were clearly stated by the faculty and often differed significantly from those which computing personnel or administrators might have selected or thought about. Fourth, duplication of resources was avoided by determining which areas had the same hardware and software and could share expenses. This in turn facilitated campus-wide planning. Fifth, vendors were given an idea of the resources which faculty and staff considered important. Last but not least, the document produced by the Task Force gave the administration a tool for incorporation into campus strategic planning and for blueprinting future computing growth patterns.

Although this method of strategic planning was used within an academic environment it could be adapted for use in a company. The trend today is for businesses and educational institutions to parallel each other. The concept of using a small team who are a cross sample of the whole organization is a viable alternative to the traditional one in which administrators from the computing area make the decisions about resources that they believe their consumers want. Our experience showed clearly that our consumers had different needs, emphases and priorities from those defined by computing administrators. By including these faculty perceptions in a strategic plan, our consumers have the feeling that they have "bought into" the five year academic computing plan.

Table 1

SURVEY USED BY THE ACADEMIC COMPUTING STRATEGIC TASK FORCE

This survey is intended to develop a picture of the long term needs for computing at UTC. It will be used to help construct a 3 - 5 year strategy for the computer area. Answer the questions both in the light of your needs and the needs of colleagues in your discipline and of your students. Please use additional sheets to answer these questions.

1. Please provide the following demographic information:
   a) Self-rating - Are you a heavy, average, occasional or non-user of computers?
   b) Do you own a personal computer? If so, what type?
   c) How long have you used computers?
   d) What is your academic unit?

2. What long term changes would you make to improve access to mainframe computers and/or micro clusters? What additional types of computer access to the library would be most helpful?

3. What computer resource improvements and additions (hardware, peripherals, software) are critical for: a) Your research? b) Classroom teaching?

4. What types and levels of training and troubleshooting should be provided to faculty and staff by the UTC computer staff?

5. What would you consider to be appropriate university standards for the purchase of software and hardware (particularly PC)? If appropriate, at what level should standardization occur? Department? College/School?University?

6. What should our policies and procedures be on acquiring, maintaining and replacing hardware and software?

7. What should be the campus objectives for electronic telecommunications including electronic mail?

8. How can new technologies such as CD-ROM, Videodisc and Hypertext be identified and incorporated into the curriculum?

9. How can computer usage be best integrated into the curriculum across the campus?

10. What rewards are required to encourage faculty development of courseware and faculty incorporation of innovative computer use into instruction?

11. What public services involving the computer are on-going but under-publicized at UTC? What are appropriate directions for future computer-based public service efforts?

12. In order to maximize available hours of access, what are appropriate security arrangements for our micro computer labs, clusters, and their software?

13. What single change in your computing environment would most encourage your productivity?

14. What other comments would you like to make on the academic computing system at UTC and/or on a 3 - 5 year computer development strategy?

TABLE 2
MATRIX OF ANALYSIS OF RESPONSES

ACCESS	Pct. Grps	GROUP 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Sum	Pct Resp
Micro/terminal in every office	79		1	2	1		1	1		5	2		3	2	2	1	1	1	2	7	32	45
More evening/weekend access	74	3	1	1			3	1	3	3	1		2	1	1	4			1	5	30	42
COMMUNICATIONS/NETWORK'G																						
Campus-wide network	84	3	1	3		4	4	5	2	4	1			1	4	3	1	4	1	7	47	66
Email in each office	90		1		2	2	2	1	1	3	2	1	3	3	4	2	2	1	1	5	36	51
COMPUTER LITERACY																						
Computers across curriculum	39	1					1			2			1			1			1	3	10	14
Separate short course (students)	26			2	1		1							1						3	8	11
CONSULTING/TRAINING																						
Hardware/software support	95	1	2		1	1	3	3	3	4	1	2	2	3	1	4	4	3	2	6	46	65
Short courses/seminars	79	3	2	4	2		2	4	4	4			1		1	1	1	2	2	7	40	56
FACULTY SUPPORT																						
Released time	90	3	2	2	3	1	2	3	4	2	3	1	3	1	2	3	3		2	5	45	63
EDO recognition (softw. publication)	74	1	1	2				1	2		2		3	3	3	3	3	2	1	4	31	44
INSTRUCTION																						
Electronic classroom facilities	74	2			3	2		2	3	4	2			3	1	3	3	1	1	5	35	49
Access to new instructional tech.	47	2	1	1		1	1		1		2						1		1		11	16
HARDWARE/EQUIPMENT																						
More printers/plotters	74			2		1	1		2	2		2	3	1	3	2	1	1	1	4	26	37
State of the art equipment	58	1		1		1		1	1				1	1		2	2		1	2	14	20
LIBRARY																						
End user search (CD-ROM, online DB)	79		3	3	1	2		1		4	3	1	2	1	4	1	1		1	3	31	44
More ports to library	42				1	1	1						3		2		1		2	3	14	20
POLICY AND ORGANIZATION																						
Plan for obsolescence	74	1		2			1	1	1			1	4	2	1	3	1	1	1	7	27	38
Separate computer budget line	68		2	2	1	1			1	2	1		1	1		1	1		3	3	20	28
PUBLIC SERVICE																						
More public access to computers	32				1		1			1			2		2	1					8	11
More media cov. of acad. computing	26	1														1	1		1	2	8	9
SECURITY																						
Staff each micro cluster	95		1	1	3	1	2	2	2	3	1	1	2	1	1	2	1	2	2	5	33	47
Lock down micros	37		2				2				1				1			1	1	3	11	18
STANDARDS																						
Support selected configurations	58		1	1	4	3	4	2	1	1							1		1	3	22	31
Department sets standards	58	1	2	2					2	1	2				2		2	3	2	1	20	26
SOFTWARE																						
Software review mechanism	47				1			1			1			2	2	1	1	2		2	13	18
More micro pkgs. from university	47	2	1	3					1				1	1		1	1		1		12	17

* Note: Only two top-rated items in each category are shown.

TITLE: What is a CSL and What Do I Do With It?

AUTHOR: Presented by: John Bishop

For more information, please contact:

INTEREX, 585 Maude Ct.

Sunnyvale, CA 94086 (408) 738-4848

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

(Handouts available at presentation.)

PAPER NO. 5002

TITLE: Improving the Effeciency and Accuracy of Your Information Capture Through Automated Data Collection

AUTHOR: Ray Agrusti (201) 492-0990
Eagle Consulting & Development Corp.
170 Kinnelon Road, Suite #3
Kinnelon, NJ 07405

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

(Handouts available at presentation.)

PAPER NO. 5004

An Integrated Voice/Data System for a Small Staff

Charles R. Williams
Beloit College
Computer Services
700 College Street
Beloit, WI 53511

608-363-2458

BACKGROUND

Beloit College is a small liberal arts college in the Midwest. We have had an enrollment of about 1000 students for many years. Our computer services staff size has remained constant over the past eighteen years. Over that period of time, we have seen changes in the roles of the staff members and a tremendous growth in computer usage throughout the campus. During this same time frame, we have moved from an operator controlled plug-in switchboard to an on-site PBX system. This was all done with no increase in staff size, but rather reallocation of duties of existing staff. The result is that we had a staff of five, plus students, for Computer Services and a staff of three for communications. This is a slight over statement, as the communications staff consisted of the Director of Security, his assistant, and his secretary/dispatcher.

In 1971, Beloit College's computer system consisted of an IBM 1800 computer. The 1800 was a batch oriented machine that required all the input and output to be controlled on site. We made modifications to the operating system to allow us to use slow (110 baud) teletype terminals. These communicated via the normal dial-up campus phone system. In 1980, we installed our first HP3000. In going from a batch oriented machine to one that is a transaction oriented as the HP3000, we had to change our philosophy as well as our data communications system.

Data communications was handled at first by direct connect terminals. We soon reached the state that this was not adequate and installed multiplexers to handle the increased load over on-site twisted pairs borrowed from the PBX system. As we expanded to still other buildings and the buried wire was not adequate to handle all circuits directly to the computer room, we piggy backed multiplexers to multiplexers with short null modem cables.

PROBLEMS WITH EXISTING SYSTEM

This situation led to many problems—both on the voice side and on the data side. Needless to say, the system was becoming so complex that we were having difficulty maintaining the data communications side of the system. We needed something simpler yet more comprehensive.

On the voice side, we had several problems. They included:

1 - No growth;
2 - Lightning damage;
3 - Blocked trunks;

4 - High maintenance costs;
5 - Slow repair response and high cost of on-site repair inventory;
6 - Excessive downtime;
7 - Limited flexibility;
8 - No student room access;
9 - Limited intercom capabilities;
10 - Difficulty of adding or changing multi-line sets;
11 - No First-In First-Out (FIFO) management of incoming calls;
12 - A difficult billing system; and
13 - Lack of the ability to transfer incoming 800-number calls.

I will discuss these items in three groups—(a) physical problems; (b) maintenance problems; and (c) logical problems. The first group consists of items 1, 7, 8, 9, and 10 from the above list. We had reached the maximum size of the physical cabinets of our existing PBX. This meant that we could not add additional lines, particularly to residence halls. Residence hall room phones have become an important recruiting advantage in the search for students. We were unable to take advantage of this without replacing the PBX. The age and limited software capabilities of the PBX severely limited our flexibility in what we could do with the switch to improve on campus capabilities. Any intercom groups or multi-line occurrences required expensive multi-button telephone instruments and 25-pair wire. Additionally, the intercom groups had to be within the same building because of the limitations of the key sets. This was a difficult circumstance for us, as we have some departments spread into multiple buildings. Thus, from a strictly physical constraint perspective, we needed to do something about the PBX.

From a maintenance standpoint, we were not much better off. The items involved here were numbers 2, 4, 5, and 6. We were limited as to whom we could go to for maintenance because the PBX manufacturer would only sell replacement parts to a limited clientele. We were, thus, caught by the prices these service providers were charging. This led to a higher cost of maintaining the PBX than we were led to believe when we purchased it. Our location did not make it possible to have a service representative on site and our service representative covered much of the state. This led to long lead times for emergency service calls because they often involved lightning damage and the same storm that affected us also might affect other customers.

In order to alleviate some of the response time problems, we kept an inventory of commonly needed parts on site. This was at our expense not the service provider's. Finally, with only one service representative and the limited spare parts stock both with him and with us, we experienced an excessive amount of down time just to accomplish normal repairs and upgrades. This did not include the downtime we suffered as a result of power outages to the building housing the PBX. We clearly needed to do something to control maintenance costs and to make maintenance more responsive to us.

The logical problems with the switch involved items 3, 11, 12, and 13 from the above list. We had a limited number of incoming trunks. At busy times, all the trunks would be busy, thus leading to a blocked trunk phenomenon. This meant that no additional calls could be made either incoming or outgoing. We did not want to add trunks because our switchboard operator was already overworked by the current number of in-

coming calls. The PBX did not have direct in dial capabilities; therefore, all incoming calls had to go through the operator.

The operator's workload led to another problem—a constantly ringing phone for some incoming calls. The switch and the trunks were not configured for FIFO management of incoming calls. This meant that a person who called in on a trunk far down the hunt sequence could sit with a ringing phone for a long time while the operator answered calls that kept coming on trunks higher in the hunt group that were freed up before the waiting calls could be answered.

Billing for long distance calls was very difficult and essentially had to be done manually to assure accuracy. We developed our own hardware to capture the Station Message Detail Recording (SMDR) information from the switch. This data was then fed to a dBASE date base on a PC. When the long distance bill arrived, there was no automatic way to match the calls. We, therefore, billed long distance calls based on the average cost per minute of all calls. Thus, all long distance calls were billed only by time and not by distance.

Finally, the switch was not equipped to handle incoming 800 number calls. In both our Admissions and our Development programs, we make extensive use of 800 numbers. Not being able to transfer the 800 number calls was extremely poor public relations because it meant that, if the person being called was in a different department than the department receiving the call, the final recipient would have to make a return call. Both of these situations needed fixing.

The problems on the data side of the equation included:

1 - No flexibility;
2 - Separate wires for each host;
3 - No interconnectivity;
4 - Lightning damage;
5 - Rapidly increasing host population;
6 - Equipment from multiple vendors;
7 - Expensive step costs for expansion and limited or no expansion capabilities in some areas; and
8 - A modem per machine situation.

Our flexibility was limited by the existing wiring scheme on campus. This wiring scheme was designed with only voice communications in mind. The addition of data to the wiring scheme presented us with severe capacity problems. Also, within buildings, we were required to do all our own cable laying. This became a very labor intensive state that stretched our small staff to its limits many times. When people wanted to move a terminal, it meant that we had move the wire, which often meant going through concrete floors.

As we added new hosts, we needed to allow users to access multiple hosts. This meant a wire per host and a switch box at the remote end. This is again labor intensive and expensive. With the planned addition of two hosts that would require campus-wide access to each, we knew that some sort of data switching device was going to be imperative.

We had no interconnectivity among our various PCs and MACs except for some building specific local area networks (LANs). This meant that files were transferred via sneaker net, a very slow and cumbersome process involving the carrying of floppy discs. Also, file transfer between types of machines required special equipment that was only available in a few locations. We needed to change this situation.

Our data communications equipment suffered the same lightning damage problems as the voice communications system. This became a problem from both the prospect of having enough spares on hand; to getting the multiplexers, terminals, PCs, or other equipment repaired; to doing the replacement work. We needed something that would eliminate or reduce the damage, which was simple to repair, and which had an impact on a minimum number of users. The cost of repair also was getting out of hand and our insurance company was looking very carefully at our claims.

We also were suffering from a modem per machine complex. In some ways this is not bad; it provides a great deal of redundancy. However, it increases costs considerably and the modems used are a minimum configuration. There is no economy of scale for the modem per machine system. By creating a central pool of modems to which all users could have access, we could use a better quality modem (e.g., higher speed) because fewer modems would be shared by the large user population. This is possible because, at any one time, only a few people would want to use a modem. If we could separate the modem from the telephone, it would be a great advantage as well. With a modem in the office using the telephone line, one is no longer able to use the phone for voice calls while using the PC to communicate over the modem. In an administrative office, this is simply not acceptable.

We had no capability for expansion and we were seeing a tremendous growth period in the near future. We were planning on automating the library card catalog. This step does not make sense unless the catalog is available from many different locations, particularly those places in which people work and study. It also helps to have the library catalog available during time when the library is not open. This means that access to the library computer must be available outside of the library. We had no way of accomplishing this without adding new wire or changing our data communications system entirely.

We were also looking towards the implementation of a statewide network, WiscNet, which would give easy access to the Internet. Again, we had no way of making the server for WiscNet available without either rewiring or installing new data communications equipment.

SOLUTION

Although some would say that we did not choose the "best" solution, the solution we chose was the best that we could come up with for the price in the time frame available and that allowed us to implement the system without any additional staff. We chose to install a Centrex telephone system with enhanced voice and data features provided by ACOIM (Ameritech Central Office Information Management). While none of what I have to say should be considered a sales presentation or promotion for the particular equipment that we bought, I think that you must know what that is in order for me to be able to discuss it sensibly. The advanced voice and data features are being provided by equipment from David Corporation.

Why did we make the decision that we did? I had been looking at various data switching devices before we chose to change the voice telephone system. In almost all cases, they required a substantial initial investment, a separate maintenance contract from the telephone PBX, our own in house wiring scheme (either shared with the voice system, completely separate, or a combination), potential additional staff, and many of the existing limitations that we were trying to avoid.

If we had been planning to retain the existing voice PBX for a considerable period of time, a separate data switch would have been a good choice. When we looked at the proposal from Wisconsin Bell, we realized that it solved many of our problems at a cost of initial investment close to that of a data switch alone. The most important problem it solved was that of additional staff to support the system. By contracting for services rather than equipment, we eliminated many of the management problems of on-site equipment.

How does this system solve the problems associated with our existing voice telephone system? The problem of incremental growth is eliminated. We are paying for individual lines and can add or delete them at will. If our requests cause the telephone company to add equipment, the cost is built into our existing rates. There are no step increases of n>1 to get just one additional line.

The physical limitations of our voice PBX are eliminated. Our continual problems with lightning have at least been moved to the telephone company. They are now responsible for the switch, the one item that has borne the brunt of the lightning damage. We are responsible for the instruments but, they have not been damaged to any great extent. We also are able to take advantage of the telephone company's greater skills in designing protective circuitry.

We will no longer experience blocked trunks. With Centrex, each user has his or her own telephone and external line. The only time a caller will receive a busy signal is when the particular line is busy. The direct in-dial feature of Centrex also adds a feature that which eliminates the need for long distance callers to begin paying toll charges as soon as the call is received by the College, even if the desired party is not available.

Those calls that still come to the switchboard will be handled in a FIFO manner due to the new trunk features that will be installed. The direct in-dial feature also will substantially reduce the workload of the switchboard operator therefore reducing the time callers will wait before being answered.

Our maintenance problems have been simplified by moving the inventory expense to the telephone company. The only local equipment that we have to maintain are the internal wiring and the instruments themselves. We have no large inventory of spare parts. We do retain some spare instruments and data adapters, but the cost is small in compared with the inventory that we were maintaining before.

We pay the price of leasing the lines and the Centrex service. We thought that, in our circumstances, this was the best trade off. We have improved the response time on repairs considerably. No longer are we the victim of a small service staff; we have the entire telephone repair system behind us. We also have access via the switch to certain

maintenance functions ourselves. This allows us to make some simple diagnostic checks before calling the repair service.

By using Centrex with a digital add on, we have simplified things like residence hall access, multi-building intercoms, multi-line sets, and the transfer of 800 number calls. We are able to take advantage of dormitory Centrex to expand service to the residence halls. This service is cheaper to the students than if they installed their own phone and allows them to access some of the data switching features if they want to do so.

Multi-building intercom access becomes a reality with this system because the switching is centralized. Because of the digital nature of the system, every phone becomes a potential multi-line instrument.

By in-house programming of the switch, we are able to add lines to existing phones and to change features programmed into a particular button of a telephone. Again, the Centrex switching facilities will permit us to allow or disallow on a per number basis the ability to transfer an incoming 800-number call. This will greatly improve our image to the public.

One final advantage is that we will receive machine-readable billing statements from the telephone company. This will allow us accurately to determine the costs for each long distance call that is made. This will, in turn, permit us to bill the costs back to the individual departments accurately.

The data that we will be able to collect will allow us to determine which numbers and lines are being either over-used or under-used. This will be a great aid to department heads in the budget planning process. Receiving the billing information in a machine readable form also will eliminate much of the manual work that has been performed just to distribute an inaccurate internal bill.

We have derived similar benefits from the perspective of data communications. We now have a uniform wiring scheme throughout campus for both voice and data communications. We are no longer stringing our own cable, bound by the limitations of the number of wires within or between buildings. The wire was installed with growth in mind. We have as much connectivity as we desire, from none to total, and we have control over the connectivity and access. The data switch is user name and password protected and each user can be controlled as to what he or she may access.

Our expandability and flexibility are considerably enhanced. With little effort, we can install a microcomputer or terminal wherever there is a telephone. On the other side, we can install new hosts as they come into being and make them accessible to the users simply by adding the host name to the data switch, providing some access ports to the host, and advertising its existence. No longer do we need to be concerned about how many hosts need to be connected to which terminal devices. They may be connected to all that are appropriate. We have, in the same way we did for voice communications, moved the responsibility for lightning protection from ourselves to the telephone company.

DESCRIPTION OF SYSTEM

The system that we installed was an enhanced Centrex system from Wisconsin Bell with additional capabilities added by a David Corporation add-on facility. This came packaged from the telephone company as IIN and ACOIM (Integrated Information Network, Ameritech Central Office Information Manager). The telephone company did all the internal and external wiring.

The internal wire will become our property and we will be responsible for its maintenance. The outside wire will remain the property of the telephone company. Thus, keeping track of all the cable pairs and splices will be their responsibility not ours. The maintenance of the switch devices will be theirs as well. We will be responsible for maintaining the individual instruments and data adapters. This gives us a great deal of relief from the existing maintenance problems at a budgetable cost, including the costs of additional equipment and lines.

On the voice side, some of the features that were of interest to us included:

1 - Automatic hold between lines on the same phone;
2 - Automatic line preselect when the handset is lifted;
3 - Call pick-up with the identification of the ringing telephone available to the answering party;
4 - Call timing;
5 - Configurable auto-dial numbers;
6 - Delayed ringing to another line or set for secretarial coverage;
7 - Differential ringing so that several phones in the same area can be distinguished from each other;
8 - Exclusive hold so that someone on another instrument cannot pick up a call that has been placed on hold by another person;
9 - Multiple intercom groups across building borders;
10 - On-campus caller identification;
11 - Last number redial;
12 - A programmable message lamp simplify the addition of voice mail;
13 - Multiple appearances of a single line on different phones;
14 - Multiple lines per phone;
15 - Privacy and privacy release;
16 - Single button access to Centrex features; and
17 - Speed dial.

I will discuss some of these features more fully. The call pick-up with identification will allow a person with a display phone to pick up a ringing phone in his or her pick-up group and know whose phone is being answered. This makes for a more business-like response when answering. The delayed ringing for secretarial coverage allows secretaries to cover for each other without listening for each other's phone to ring. When one line rings and is not answered in a selected number of rings, a second phone starts to ring with a distinctive ring. The first phone continues to ring and the call goes to whomever answers first. The second person will see a display of whose phone is ringing and thus can answer with the appropriate greeting. The privacy and privacy release features will allow us to eliminate multiple people answering the same line within seconds of each other but also allow another party onto the same line as appropriate.

Some of the data communications features that were appealing to us included:

1 - Autobaud capabilities on all ports at user option;
2 - Data call hold;
3 - "Dial" data connections;
4 - Dedicated connections;
5 - Hunt groups for data calls;
6 - Menu prompting;
7 - Password access;
8 - Port contention;
9 - Port queuing;
10 - Simultaneous voice and data over the same twisted pair;
11 - Speed matching;
12 - Timeouts; and
13 - A modem pool.

I will discuss each of these features, as they are important additions to our existing system.

The autobaud on all devices allows us to pick and choose our data configuration for optimal performance. For those devices that should operate at a fixed speed, we configure them that way. For devices that need to run at various speeds, such as modems, we can configure the switch so that it can determine the current speed of the device. This is a considerable improvement over manual speed matching.

The ability to put one data call on hold and to make a second data call is immeasurably useful to people in support roles, such as programmers. It allows one to be doing work in one session or on one machine and then to answer a question that would require access to a different program or system and then resume the first session at the point of interruption. This makes the support person much more responsive than if he or she had to take the time to save the current work, log off the current session, and then initiate a new session.

The ability to either dial or dedicate a connection gives us a great deal of flexibility. The dialed connection means that each user has a list of allowed hosts and may connect to any host at any time. This eliminates the need for multiple wires to multiple hosts. The dedicated connections allow us the ability, for example, to connect a printer to a particular host without user intervention. It also allows us to limit users at particular terminals to a single host.

Menu prompting simply makes the system easier for the user to work with and thus simplifies support. This, in conjunction with hunt groups, port contention, and port queuing, provide considerable ease of use and ease of support. Hunt groups allow multiple data ports to be given a common name so that the user only has to know a single name to access the multiple ports available on a given host. The port contention allows us to use fewer ports than potential devices that would be connected if each user had a dedicated port. This is possible because many users are logged on for only a short period of time.

The port queuing feature allows people to wait in line if all the ports are busy at any time. By using the data port usage statistics that are available in machine readable form, we can determine if there has been too much queuing or too little access of ports. This will allow us to control our expenses for data communication with documentation to back up our decisions.

Password access and timeouts are security features that improve security at little expense to the user and almost no cost to management. Each user is given a password and user name uniquely to identify the user. Then, all access through that user name may be tracked and security problems can be brought to light. The use of a user name allows us to let the user take privileges to the terminal wherever it may be located. If a security problem exists, a user can change his or her password to reduce the chance that an outside user can gain access. Since these passwords are in addition to normal HP3000 or other system security, they are an additional level of protection. The time-outs are used to help control access to a host for which no work is being done. By setting a timeout value on the host ports, we are able to force a log off of a user that is not doing any work.

Simultaneous voice and data over the same twisted pair, speed matching, and a modem pool are all user convenience features. The modem pool will allow us to use higher grade modems than we would use if we installed one in every microcomputer that needed one. This is a result of the economy of scale that the pool brings. After all, one uses a modem only rarely and it should be available to someone else when one is not using it. Pooling the modems allows this. The modems we installed are autospeed sensing from 1200 baud to 9600 baud. This allows great flexibility in making data calls in both directions.

The speed matching feature will allow the user to retain the current speed setting that is used for internal work and still make an outgoing data call regardless of the speed of the receiving modem. The data switch will allow the modem to sense the receiving modem's speed and then to change to that speed. If this is not the same speed as that of the user's terminal or micro, the data switch will convert so that the user does not need to change the local speed setting.

By using the modem pool and the simultaneous voice and data feature, a College staff member may be making an outgoing data call through the modem pool while talking on the telephone at the same time. This is not available when one is sharing the same line for both purposes with an analog network. This is of considerable importance both from a business function standpoint as well as from a convenience standpoint. How many of us have ever tried to exchange files with someone from another location and wanted to talk to the other person at the same time as we are trying to establish the data connection? This makes such things possible.

WHY CAN THIS BE DONE BY A SMALL STAFF?

From the above comments and descriptions, it sounds like this is impossible to support with a small staff. This is not the case. We do not plan on adding any staff to support this system. In fact, some of the functions are being added to existing staff. This is possible because of the simplification of the tasks and because this system is removing some functions from those same staff members. Much of the staff overhead has been transferred to the telephone company. We are allowing them to do what they do best—

provide communications services. We are retaining those functions that we do best—managing our own resources.

Many of the changes to the voice or data configuration can be made by College staff through terminal access to the switch. This will eliminate much paperwork and liaison with service providers. The standardized wiring of all voice and data devices makes supporting this much easier than our prior system of specific wiring schemes for individual devices.

Most of the difficult work is in the set up of the configuration of the switch. This is a one-time staff expense that will be recovered in short order by the efficiencies of operation. The voice configuration required us to survey each existing phone and to determine what should replace it. The telephone company then converted this into documents that were used by their data entry personnel to create the initial configuration. We will only be responsible for changes and additions.

The data configuration involved more work on our part to set up because it is more complicated than the voice system. The process was essentially this:

1 - I made a list of all "users," terminals, microcomputers, and hosts.

2 - I determined who should talk to whom, i.e., which device could communicate with which other devices.

3 - I created port groups, similarly oriented users.

4 - I configured the port groups with their appropriate characteristics.

5 - I created feature class of service groups. This essentially described what features of the data port a given user could change.

6 - I created access class of service groups. This defined which users could have what kind of access to which hosts.

7 - I configured the data adapters as to the port characteristics to which they were attached.

8 - I created default user names and passwords and determined the initial lists of which users could access which hosts and which hosts would allow which users to access that particular host.

I found that using a spread sheet program to hold all this data was a great aid. I was able to look at the data in different arrangements. I am using the spread sheets as the basis for my local documentation regarding the switch configuration. I then transmitted this information to the telephone company and they did all the initial data entry and validation. I will only be responsible for additions, deletions, and changes from this point on.

The final change that makes this easier to support was in the wiring for the individual devices. In order to make moves easy, we wanted to eliminate the need to carry a soldering iron with us when we went to move a terminal. What we did was to use DB-9 or DB-25 to RJ-45 plugs that are then attached "permanently" to the serial port of the microcomputer or terminal. All jumpers and other unique wiring for the device is within the hood. The wires from the DB end then go to consistent location at the RJ-

45 end of the hood. We are then able to use a straight through 4-twisted pair wire to connect any two devices. This allows a move to take place simply by disconnecting the RJ-45 plugs at one location and then inserting them at the new location.

This paper is being written as the installation of the equipment and software is proceeding. The change over date is early in January. During my talk, I will be able to discuss how the actual implementation went; what problems we encountered; what good things happened; and what recommendations I can give to others who are embarked on a similar project.

FACILITY PLANNING

Norman A. Hills, President
N. A. Hills Computing Services Limited
336 Piccadilly Street,
London, ON Canada N6A 1S7
(519) 672-1731

CAPACITY - WORKLOAD = MARGIN

In the spirit of Mr. Macawber from David Copperfield, a positive residual for MARGIN is happiness, but a Negative MARGIN is misery. Because the equation describes a DYNAMIC situation, Margin will fluctuate and go negative at times of peak activity. At the times when it does go negative, it will be misery for the users, misery for the System Manager and misery for whoever selected the present system. As the misery continues, pressure will grow for the managers to implement some correction.

The purpose of this paper is to explore some of the elements that are important ingredients of our simplistic equation to prescribe continuing happiness.

LOCATION

The selection of a location may be influenced by a desire to have the computer in a very prominent location as a corporate showcase. If this is the purpose, then perhaps the company would be better served with a viewing window to allow visitors to watch coloruful flashing lights and rotating tape reels all mounted on neat rows of empty cabinets.

Witht he present trend to smaller and smaller units with fewer activity lights, computers are becoming much less impressive to look at and there is very little visible evidence of the large amount of internal activiy. All of these trends are making it less and less beneficial to use the computer as a status symbol, prominently displayed and centrally located within the corporate offices.

There are many good reasons to have the computer REMOTE from everything else.

1/ The protection against LOSS OF INFORMATION, which could be crippling in the event of any natural disaster, is better if the electronic storage on computer discs is several miles away from the office records that exist on paper.

   We advocate about 10 miles as an optimum separation between office and computer. This is close enough for communication charges to be reasonable and far enough for there to be little risk of both locations sharing the same natural disaster.

The compass direction of this separation should contemplate the probable paths of any natural disaters and avoid either flood plains or lines of siesmic faults.

2/ Separation between the computer staff and the general clerical office staff reduces the risk of salary comparisons between these two groups of quite disparate skill levels.

3/ This separation will also encourage more formalized communications and a reduction in the interruptions that can be such a detriment to concentration on the computing task.

SPACE

It is important to perform the SITE REVIEW with the CE to ensure that all of the requirements for space and utilities will be adequate to meet the needs of the computer. In spite of the fact that newer computers will be smaller in size and will require less power and not as much air conditioning, be sure to provide an adequate margin for all of these requirements.

The area in square feet that you provide, should be at least double your present documentable need. You will definitely face a need in the near future to add additional disc drives and it will be nice when the time comes for an upgrade, if you will have the room to be able to continue to operate the old while the new is being gradually phased in.

More important than the cost of making future alterations to your cpmputer space, will be the difficulties you would encounter in attempting to make alterations without violating the computer's environmental needs for cleanliness and climate control. Every time we have increased our physical space, we have soon regretted that we did not make it larger than we did.

Air Conditioning equipment must be sized so that it will continue to be adequate as it ages and becomes less efficient. In addition to this, it MUST BE DUPLICATED so that you will be able to continue operations while waiting for service or repairs. As this equipment ages, you should also investigate the availability of spares or standby units that could be obtained quickly to replace an aged unit when repair becomes impractical.

HARDWARE

The configuration that you purchase should in every aspect be capable of at least double your present documentable needs. This will provide room to grow from ths planned 50% utilization of capacity before you suffer the pain of negative MARGIN in our eqaution for happiness.

At the time of your FIRST COMPUTER ACQUISITION, this evaluation of needed capacity is very difficult because there will be so many unknowns to make allowance for. Depending on your degree of uncertainty, you may feel more comfortable with a larger initial margin.

The higher cost of a larger unit can be offset by opting for USED, PRE-OWNED or NEARLY NEW depedning on which title you prefer. The risk associated with 'used-cars' is not a factor here, because the purveyor of your maintenance contract assumes all of this risk. In order of declining costs, you have the following options:

1/ REMARKETED units from H-P for which the purchase price includes installation, warranty for 90 days and no problem appending this added equipment to any existing service contract.

2/ THIRD PARTY which will usually have a 30 day warranty, but H-P will insist that they must do the installation if it is to be incorporated into any equipment covered by their maintenance contract, and any equipment failure within the first 30 days creates a problem of having to assign the responsibility to the appropriate vendor, and have them provide a prompt resolution without inappropriate penalty to the user. The purchaser in this option must be satisfied that the monetary savings are sufficient to compensate for any contingency expenditures of time or money that may be needed to resolve the divided responsibilty.

3/ AS IS from another user can be at low risk if that equipment is presently running under an H-P service contract and if you can re-install it at your site within the H-P 60 day limit to qualify for a transfer of the existing service contract.

In support of this PRE-OWNED philosophy, I should comment that our initial Series II, our Series III upgrade, and our present Series 48 have all been Previously Owned units. In addition the Series III and the Series 58 on our premises that we manage for a customer are also Pre-Owned units. The cost savings associated with this philosophy have been quite substantial.

HARDWARE UPGRADES

The newer technologies with their higher reliabilities and reduced maintenance costs are creating extreme pressures in favour of upgrading. In short, the lease cost of an upgrade could be less per month than the monthly savings in the cost of your maintenance contract and power costs. When evaluating these cost benefits of upgrading, be careful to include ALL OF THE COSTS that may be associated with your utilization of the Upgraded Hardware, as this new hardware may require services that you are presently managing to do

without. For example if you are still operating a Series III with a harware service contract but with NO SOFTWARE CONTRACT, you may be facing the addition of a significant software charge. Also because you will be upgrading to a new operating system with which you have no experience, and which may include some bugs, you should also consider including the cost for RESPONSE CENTER SUPPORT in your cost comparison between your current Series III and the suggested upgrade alternative.

Performance comparisons between different Series of the HP3000 family of computers are almost meaningless. We suggest that you be skeptical of promises that the upgrade will increase performance and dramatically improve your happiness equation. Because the hardware architectures are different, and the newer operating systems do more, any statements related to performance comparisons are only approximate generalizations.

You may be persuaded to accept a nominal exchange allowance for your present hardware. The value of this current hardware to you as a standby resource is far greater than the amount of allowance offered. If you followed our recommendations on SPACE and AIR CONDITIONING, then you would have no problem retaining your present unit and gradually phasing in the new. While you have two units operating, you can make numerous performance comparisions between the old and the new. You will probably confirm many areas where the new does in fact meet the expectations, but you will also be able to demonstrate and document instances where the new falls far short of the expectations. With this type of evidence you should have little difficulty in having the H-P labs correct the condition, of which they may have been genuinely unaware.

As an example of what we were able to acheive through this process of comparison, in Table I we have the times for the same Query Find and Report Procedure run on similar configurations with EXCLUSIVE use of the hardware.

Table I
COMPARISON OF QUERY REPORT PROCESSING TIMES

Test No	Computer and Query Version	Time for query Report to Disc File
1	Series III MPE IV C.B1.A2 with Query HP 32216B.01.02	60 Minutes
2	Series 58 MPE V G.B3.02 with Query HP 32216C.02.08	90 Minutes
3	Series 58 MPE V G.B3.02 with Query HP 32216B.01.02	36 Minutes

Test 1 used 2-7925 disc drive of 125 mb and test 2 & 3 used 2-7933 disc drives of 404 mb with the Image Data Sets similarly distributed across the disc drives.

The purpose of test 3 using the Query Version ported over from the Series III was to confirm that the deterioration was primarily related to some change to Query that had taken place between the release dates of the two versions being tested. The patch that we subsequently received adequately corrected the deficiency of QUERY C.02.08, but we wonder how many units there may be that are still performing at the slower pace because the need for this patch has not been recognized.

The above approach with its advantages and opportunities, makes more sense than does the more prevalent practice of purchasing an upgrade, and then scheduling the change over to start with the removal of the old late on Friday with the new to be loaded and ready to operate by Monday morning. While these users usually have a general feeling of contentment with his improvement in performance, they have denied themselves of any opportunity to make comparison measurements of performance.

The usual practice is for the purchaser to require the installation of any computer as soon as possible. We advocate that your capacity requirements need to be thought of well in advance and an upgrade should progress at a more COST EFFECTIVE pace as follows:

1/ As soon as an upgrade is recognized as inevitable, chose the desired computer configuration and place your purchase order. On this order indicate the delivery date but stipulate that the installation may not take place until 6 months after delivery.

2/ Cancel the maintenance contract on the existing unit. You may require 3 months notice to H-P for this to take effect, but we will assume that the delivery of your upgrade will be of this same order of magnitude. As soon as your upgrade has been delivered, it can be standing by in case it is needed.

3/ Continue to use the older unit for all your production, being prepared on short notice if required, to install the upgrade. During this interval, you will have paid for the new unit but your maintenance contracts will be ZERO COST.

4/ Install your upgrade at the 6 month expiry, or sooner if needed. Your 90 day warranty on the upgrade will commence with its date of installation, and your maintenance contract costs on this upgrade will not commence until after the expiry of this warranty. In addition to the promised reductions in maintenance contract costs that justified the upgrade, you have increased this saving by

6 months of ZERO maintenance cost at very little risk.

5/ Continue to use your older computer for several purposes, for as long as it will continue to operate WITH NO MAINTENANCE. Its many uses can include:

i) Performance comparisons between the old and the new.

ii) Development machine so that compiles will not interfere with the performance for general users.

iii) Backup resource to provide continued processing for critical applications while waiting for contractual maintenance on your primary computer.

6/ As your older computer continues to age, keep alert listening for any opportunities to acquire at minimal cost, any similar units that could be a source of spare parts. Your only costs for continued use of the older computer will be the electricity costs for the computer and its incremental requirement for air conditioning.

By keeping your older computer and continuing to utilize its capacity you have for very little cost, become a multiple unit facility. In addition to the benefits of redundancy, your overall computing capacity will be enhanced. Because the architecture of the HP3000 series have inherent bottlenecks such as Disc I/O, the throughput of 2 computers operating side by side will exceed the throughput from a single unit of one step up in capacity.

HARDWARE SERVICE CONTRACTS

Service Contract options in descending order of cost are:

1/ 24 Hour 7 Days of the week

2/ Standard 8 AM to 9 PM with 4 hour response time. This is the most popular choice because it is the least expensive option that provides service between 5 PM and 9 PM at no overimte premium for installations and preventive maintenance.

3/ Next Day Coverage may be adequate if you can manage without the computer for 24 hours. With this level of coverage, any time after 5 PM for maintenance or installation will incurr an overtime charge.

4/ Return Unit to Field Service Center. This is impractical for the computers but is a good choice for terminals.

Four hour response time sounds pretty good, and H-P typically arrive at the site in about half this time. However if your are unable to bring the system up at the start of the day, you can usually expect it to be after noon before the system is returned to your use. If this failure

has occurred at a time when there are critical operations to be run such as on a payroll day, it can be a terrific advantage to have an alternative system on which these critical operations can be run while your primary machine is down.

Terminals are not critical to continued operation because they are interchangeable and there are so many locations from which they can be spared. Also because they are reliable and seldom require maintenance, we have long ago discontinued coverage on terminals, opting for time and material for repairs if and when required.

OPERATING SYSTEM

The Operating System for the H-P 3000 series of computers is constantly being revised to add new features and increase its capability. All of these enhancements have some influence on performance and have the potential to introduce NEW BUGS into the system.

A few timing tests presented in Table II illustrate the magnitude of this growth in Size of the Operating System Software through increases in elapsed time for several typical operations:

Table II
TIME FOR TYPICAL TRANSFERS OF THE OPERATING SYSTEM
Elapsed Times in Minutes : Seconds

	Series III MPE IV C.B1.A2	Series 48 MPE V G.A2.04	Series 58 MPE V G.B3.02
Warm Start (spun down)	4:38	3:25	5:25
Cold Load	5:40	6:53	7:15
Shutdown	0:7	0:9	0:10

Our policy has been to IGNORE UPGRADES unless and until they offer some feature that has a BENEFIT THAT WE WANT. The costs that will occur after an upgrade make it unwise to take this step unless the value of the benefits will compensate for these costs:

1/ The time for your operator to install the UPDATE will be sma l, but should not be ignored, particularly if there is any risk that you may not like the new system and may have to return to the old.

2/ The new update may contain new bugs for which you may need patches from the RESPONSE CENTRE. These my not be readily obtainable without Response Centre Support, the cost of which should be included in your evaluation.

3/ The new update may place some limitations of the traditional compatability in both forward and backward directions.

WORKLOAD

The subject of workload can be quite extensive, but there are several important considerations that we should mention briefly:

1/ Restrain the use of QUERY. We have QUERY assigned to an 'E' priority but have the 'E' Queue kissing the lower end of 'C'. This gives the interactive users on 'C' priority while still letting QUERY come before STREAM JOBS that are automatically assigned the 'D' priority.

2/ With training and supervision, watch for abusers of the Query 'SERIAL READ' which can be a real CPU Hog. We have been searching for ways to prohibit the use of 'SERIAL READ' or to automatically terminate on a 'SERIAL READ' but so far the smarts of Query have circumvented any techniques that we have tested.

3/ Keep the segment sizes of all programs to a reasonable size of 4K per segment.

4/ Avoid the SORT/3000 utility. In place of this use something like a linked list to arrange the desired selection of records in their required sequence.

5/ Encourage the use of OVERNIGHT JOB STREAMS for every task from which the results will not be required until the next working day.

6/ Ensure that IMAGE DATA SETS are sensibly distributed among the available disc drives.

7/ Regularly perform a DBUNLOAD and a DBLOAD so that the PRIMARY PATH entries will be contiguously located.

8/ Conciously ensure that prgrams will select the shortest chain for all database searches.

9/ Query finds should always specify the SHORTEST CHAIN item first when searching for the intersection of argument values for more than one item.

GROWTH

Growth is an insidious form of workload that will slowly erode the performance if it is allowed to continue unchecked.

1/ Dormant files will accumulate on the system and if not disposed of will contribute to the illusion that MORE DISC CAPACITY is required. These dormant files should be regularly disposed of, and we copy to tape and record in an ARCHIVE database any file that has not been used for 9 months. If it still has not been used after 12 months it is again copied to tape, again referenced in the ARCHVE database and at this time is PURGED from the system.

   This creation of TWO TAPE COPIES before purging is another example of our desire for redundancy. This provides us with two copies on tape and database entries to assist in locating and retreiving files that need to be restored.

   The trial use of 3 and 6 months instead of 9 and 12 resulted in the need to RESOTRE so many files that had an ANNUAL requirement, that we extended to 9 and 12. Each user should develop their own policy for archival. In the absence of any policy, files will relentlessly accumulate, and eventually fill all the available disc space.

2/ Files have several levels of activity and should be distributed across the tracks of the disc drive so that the most active files will require the smallest amount of seek time for the heads.

   i) SYSTEM files such as the SYSTEM SL will be the most frequently used and will be loaded first.

   ii) If the system is typically oriented to IMAGE Database applications, then the datasets will be the next highest user of Disc I/O.

   iii) Program files that are in active use should come next.

   iv) FREE SPACE for dynamic use of current needs should be reserved next by creating a dummy file to temporarily occupy the amount of space desired.

   v) Last of all should be the Dormant files that may occassionally be required, such as the files of the TELESUP account.

   vi) After loading all of the above, purge the temporary space occupying files of steep iv.

3/ DORMANT RECORDS within a Dataset will contribute to the length of the chains, and their presence will increase the search times. A clear policy is neccessary for dormant records. If these records must be maintained on disc for easy referral, then at least transfer them to an alternatve dataset so that they will not impede the active searches.

If you fail to control these factors that will slowly erode the performance of the system, then you may be contemplating a facility upgrade long before it is essential.

We have purposely avoided any reference to actual dollars in the above presentation, partly because these are continually declining, and partly because our experience would be in Canadian Dollars for which the exchange rates would distort the significance.

" S A V E M O N E Y I N L A N D E S I G N "

" CABLE MANAGEMENT FOR DATA COMMUNICATION"

by

Aldo Falossi
Exec. Vice President
CABLE MANAGEMENT SYSTEMS, INC.

"SAVE MONEY IN LAN DESIGN"

All the roads do not lead to Rome as we have been lead to believe. The same rationale is applicable when cabling a LAN without proper planning. Not all the wire will reach destinations as you need them unless a step-by-step plan devised to assure arrival to the proper destination.

The single most important aspect of a LAN plan is to assign an individual responsible for the entire project. He should function as the focal point for everything relating to the LAN, including corporate staff, building managers, data and voice communications managers and with all the various department heads as to their current and future communication needs.

The LAN "Project Manager" should evaluate the communication building and staff needs in order to define a voice and data communication structure commensurate with the present facility and proposed growth. He should develop a chart including project task, date and individuals assigned to the specific job.

During the selection of the active network components, he should ask the vendors for specific details as to the current and future plans for hardware and software to be supported.

Designing a LAN for operation on coax, to find out that future products will run on fiber will be a costly mistake. If such details are incorporated in the plan, one can lay out the fiber media in the building (with the coax) in preparation for the hardware switch later on.

Defining data type, different products, protocols and standards, which will be transferred over the network will help determine the network load as well as the paths needed.

From discussions with the departmental people, a clear definition and understanding of all the users needs will dictate a network topology and the type of cabling scheme to use.

An office may require access to four main computers, each one using different media for communication. Example: IBM 36 Twinax, WANG Twin-coax, HP Twisted-pair and 3 Com Twin-coax. In this case you may want to wire the user's office with a media which will support the highest data transfer and use personality modules (Balun's) to adapt the respective terminals to the Main Frame. Switching the various Main Frames and peripherals to the office can be accomplished via a data PBX (manual or electronic) capable of handling all of the data transfers.

Some of the key items which must be considered during the planning stages:

Expansion: If we believe recent statistics, 50% of office personnel move yearly. The cabling scheme and implementation should have outlets in every imaginable area. In heavy data communication applications (health care, insurance, reservation systems, etc.,) the project manager may want to consider sufficient spare cables to double it's initial installation.

Installation: Depending on how and when the facility is cabled may save as much as 25% of a system cabling cost. A site survey before the network is designed will help identify problems, the proper routing, and component selection of it's runs before the walls and ceilings are closed and finished.

Maintenance: Involves two distinct areas: the devices on the network as well as the cable system itself. Research indicates that maintenance can account for 5% to 10% of the network cost.

Standards: The person responsible for the network should be aware of the standards defining tele/data communication protocols and the cabling system capable of supporting those standards, especially in a multi-vendor application.

Cost: Be sure to incorporate all of your future needs in costs today. The initial design is a small percentage of the total cost. You can provide for future expansion with plenty of users drops for a small increment over the total cost. Adding new wiring after a cabling system has been installed and operative can be very expensive.

Since we don't live in the ideal communication world where standards prevail, you need to identify all of the devices which you will be connecting on the network and how they will be connecting on the network and how they will be interfaced. Main Frames, Terminals, Printers, Modems and PC's, may need to be networked. Each may be built by a different vendor, requiring unique connectors, wires and pin-outs. If a direct connection is not available, there are other methods for getting the end device onto the network. For example a parallel printer may need conversion to serial and/or vice versa.

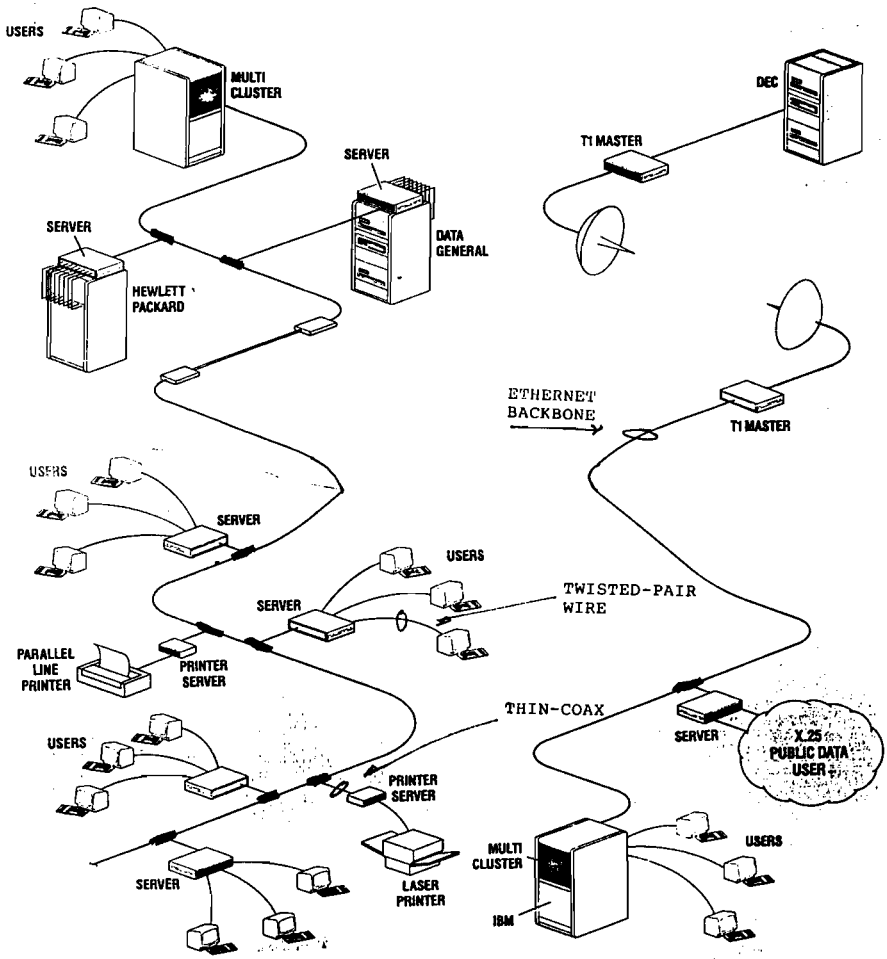
Once the applications have been defined and the connections
of the various devices determined, the topologies can be selected and a cabling system design can begin.

The system is typically divided into three (3) major segments:

Main Distribution Frame (MDF) - Sometimes referred to as backbone or broadband LAN. Typically devices connected to this network would be host computers, graphic stations, terminal servers and any other devices that need band width to transfer large data files. The medium typically used if either Fiber or Coax. Ethernet is the most widely used and accepted backbone.

User Stations (US) - The terminating points of either MDF or IDF. They normally are cabled using twisted-pair wire, although some user stations may require a combination of all media. Example: D.E.C. supports video (coax) high speed data (coax), low speed data (twisted-pair) and voice (twisted-pair) at each user station.

Intermediate Distribution Frame (IDF) - Are usually subnetworks where most data is passed locally or peer-to-peer. The media may be coax or twisted-pair wire.


USERS
MULTI CLUSTER
SERVER
DATA GENERAL
SERVER
HEWLETT PACKARD
DEC
T1 MASTER
ETHERNET BACKBONE
T1 MASTER
USERS
SERVER
USERS
SERVER
TWISTED-PAIR WIRE
PARALLEL LINE PRINTER
PRINTER SERVER
THIN-COAX
SERVER
X.25 PUBLIC DATA USER
USERS
PRINTER SERVER
SERVER
LASER PRINTER
MULTI CLUSTER
USERS
IBM

Once the network topology has been selected and the design finalized a bill of material will be generated to support the needs of the installation for today and it's projected growth.

The next step for the Project Manager is to participate in the actual LAN installation. Understanding the four (4) basic steps required for a successful job will make his future tasks (additions, deletions, changes, trouble-shooting, etc.,) a lot easier.

1) Site Survey
2) Cable System Design
3) System Installation
4) Certification/Documentation

During the site survey, an internal or contracted cable designer visits the facility where the system will be installed. He meets and works with the network Project Manager to determine the actual cable runs, layout of the Main Distribution Frame (MDF), the Intermediate (IDF) and User's drops, and to look for potential installation problems.

For intelligently designed buildings, encompassing data and/or voice, the network access points (users drops) should be available throughout the building.

Active components, such as Modems, Multiplexers, Terminal Servers, Amplifier etc., need to be installed with provision for occasional service as part of a preventative maintenance program.

The topology and installation of the cabling system, has been a direct relationship to costs and mean time between failure (MTBF). This can account for 50% to 65% of the network cost and is determined in part by the physical layout of the building itself.

Redundant and spare cables may be necessary if the facility cannot afford down time (ICU's in a Hospital). The spare cable should reach the same destination using different routes. This approach reduces the chances of something damaging both lines at the same time. The obvious drawback is cost and complexity.

Components for a typical network system based on an Ethernet (for Main Distribution) and Twisted-pair system account for about 30% to 40% of the total cost.

The information gathered during the site survey is used to make drawings so that user's drop, demarcation points, cable runs and the location of active components may be properly identified and become part of the final documentation.

A cabling system design represents 2% to 5% of the total systems cost, the variance depends on the complexity of the network as well as the duplication of like cable segments. Once the design and bill of material are completed the actual marriage between the Project Manager and installing company takes place.

The relationship will help maintain schedules, keeps labor cost down, elevates down-time and assures the Project Manager of what the system will be like and how it will be maintained. Simple matters, such as a cable run near an elevator motor, can be susceptible to noisy lines and create havoc with the network.

System certification supervision is a must for the Project Manager. Tests should be documented, and if available photographs should be taken of the spectrum analyzer at various points on the cable system. Photo's are useful in trouble shooting any future network problems that may arise.

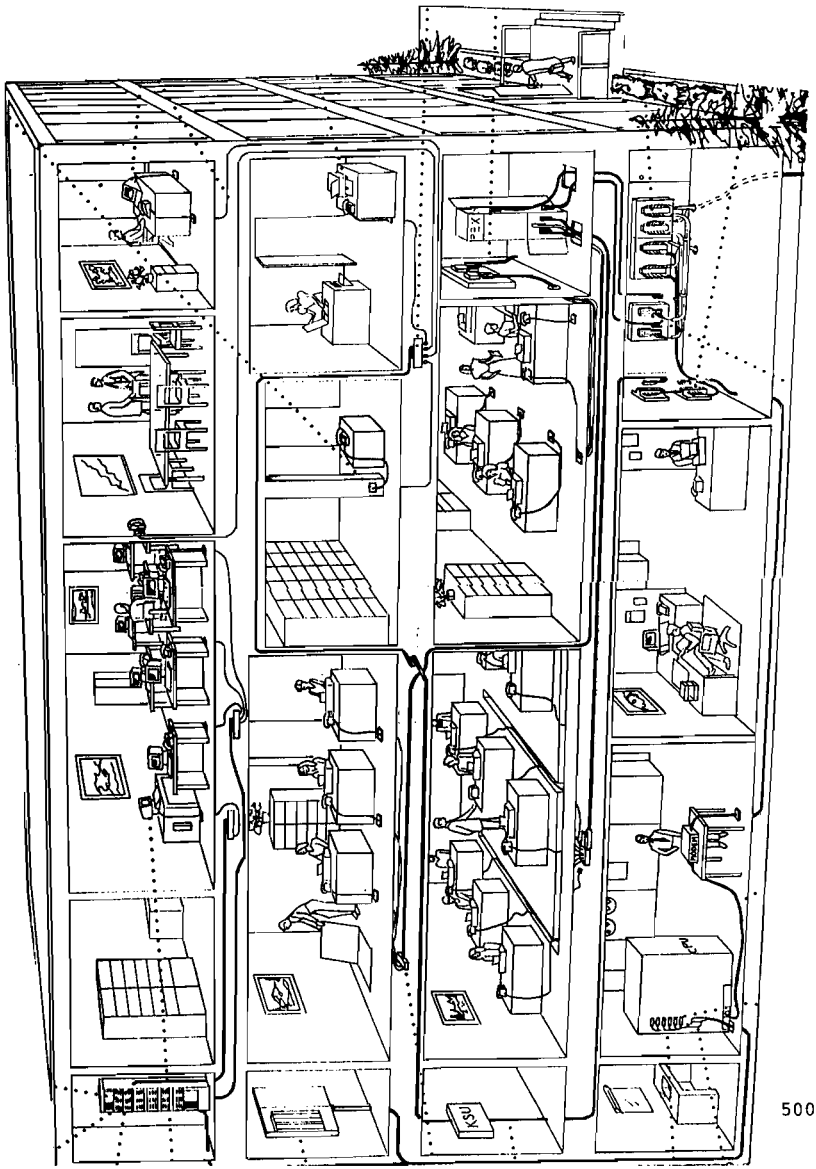
Maintenance of a LAN is another area which is often overlooked,and therefore expensive repairs can occur. Maintenance of a LAN encompass more than just repairing a broken cable.
The Project Manager must have trained people, appropriate test equipment, spares and a preventive maintenance program.

Personnel trained in communication hardware and networking are hard to find and training is an expensive proposition, especially if your LAN is for a small system. There are numerous companies which provide maintenance contracts for fixed fee, and you don't need to procure test equipment and/or spares.

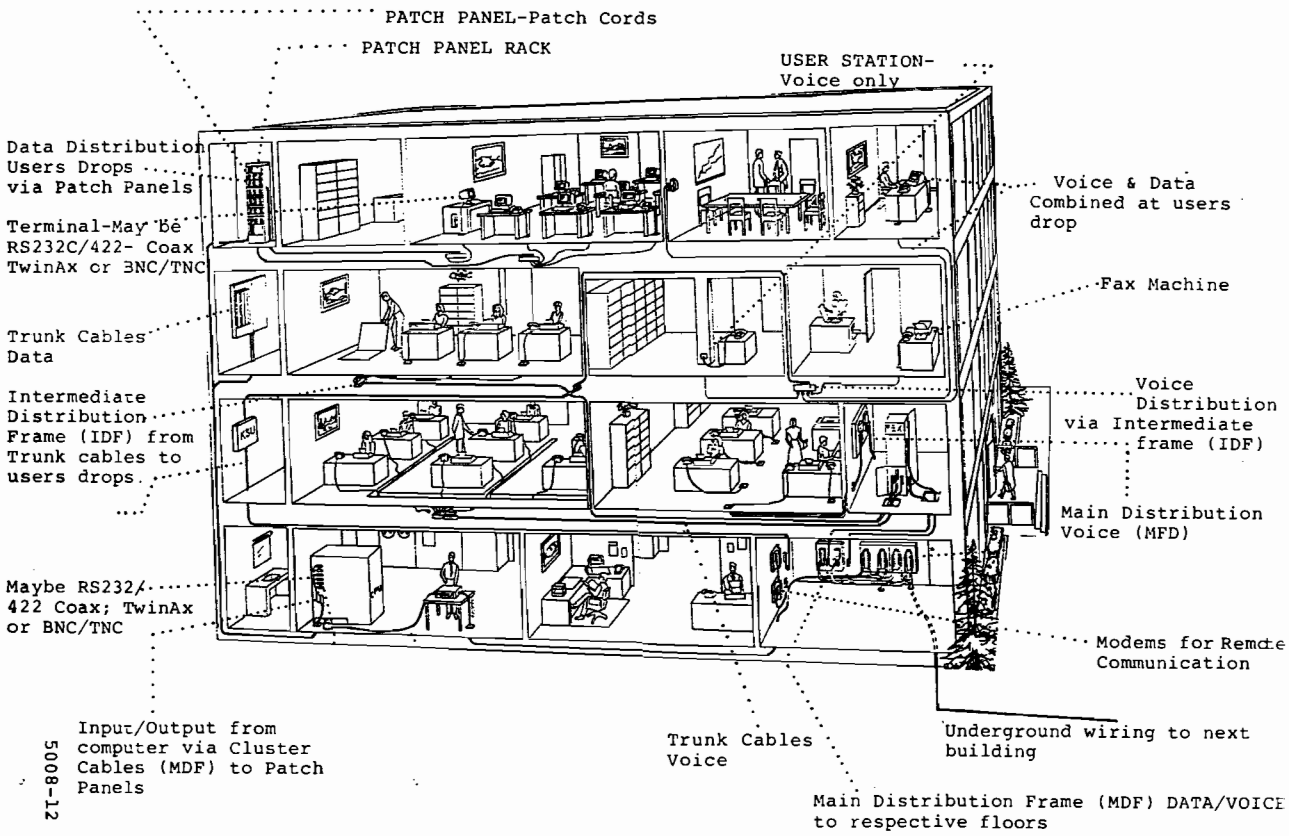
With dedication and commitment on everyone concerned your network will be well planned and the design/installation phase will be done in good business harmony with your vendors.

The network cabling system designer may make cost saving recommendations as to spares, test equipment, inside v/s outside maintenance.

Just remember how well your LAN works is directly related to the Project Manager and his dedication to the project.



5008-11


PATCH PANEL-Patch Cords
PATCH PANEL RACK
USER STATION-
Voice only
Data Distribution
Users Drops
via Patch Panels
Terminal-May be
RS232C/422- Coax
TwinAx or BNC/TNC
Trunk Cables
Data
Intermediate
Distribution
Frame (IDF) from
Trunk cables to
users drops
Maybe RS232/
422 Coax; TwinAx
or BNC/TNC
Input/Output from
computer via Cluster
Cables (MDF) to Patch
Panels
5008-12
Voice & Data
Combined at users
drop
Fax Machine
Voice
Distribution
via Intermediate
frame (IDF)
Main Distribution
Voice (MFD)
Modems for Remote
Communication
Underground wiring to next
building
Trunk Cables
Voice
Main Distribution Frame (MDF) DATA/VOICE
to respective floors

RELOCATING THE COMPUTER CENTER
(from conception, through deception, to completion)

P. Troy Jackson
Missouri Western State College
Computer Center LRC 104
4525 Downs Drive,
Saint Joseph, Mo. 64507
(816) 271-4565

Haven't you ever dreamed of designing your own computer room, your own office, or even your own work area? Think of all the changes you would make, all the corrections. How many times have you wondered (frequently at the top of your lungs) "what idiot designed this place?" At Missouri Western State College (MWSC), I was given the opportunity to live that dream, and realized that the designer of our original computer center may have been a reasonably intelligent and logical person before the project started. The original concept of a new facility began about five years ago. That may seem like a long time for those of you who do not have to deal with governments, but for those of us who do, it's about average. The actual final design, construction, and full implementation took about a year and a half.

THE WAY WE WERE

In order to fully appreciate how far we came with this project, one must first understand our humble beginnings. I arrived at the school in the summer of 1987. At that time the Computer Center was located in the basement of the Learning Resource Center (LRC) or the library for the less esoteric. The computer room itself was a 15 X 15 room sporting the ever popular "fish bowl" effect that operations personnel so dearly love. The "fish bowl" effect was popular back when people thought computers were magic and security was maintained with "authorized personnel only" signs. Three of the four walls were cinder block construction. The fourth was an eighth of an inch thick portable retainer supporting a glass (no, not plexiglass) window and a wooden door, with a combination lock. Security was critical. The floor was raised, well actually sunk, three feet and covered with tiles which allowed for ventilation, wiring runs, storage areas, trash disposal, and a rather large dust bunny breeding ground. The room was equipped with a haylon fire suppression system, an environmental control system, and a power conditioner for the majority of the equipment. Data distribution was obtained through a composite of direct connects and muxes linked in every conceivable manner, most of which where unidentifiable by anyone living. The result of this wiring montage was basically chaos. We had wires going everywhere and no idea what was what.

At that time the original proposal for a new facility had already been submitted. As is typical in public

organizations, the initial plan did not address the computer room, but rather called for a larger LRC. It was only when someone asked if the construction would effect the Computer Center that I actively became involved. From the onset, a massive re-education project was necessary. The plans called for knocking out two cinder block walls and moving the computer room eight feet south to allow for a hallway, all to be accomplished without shutting the system down. It is said that my objections were heard at the highest levels (200 to 400 decibels). Being in charge of both Operations and Security, I was somewhat concerned that both would be jeopardized with such a plan. I became extremely dogmatic in presenting reports that indicated that the systems would have to be shut down during any times when the environmental conditions (temperature, dust, combustible fumes, or vibrations) could not be maintained. In light of my concerns, it was determined that I should design the computer center portion of the plan.

As managers, it is our responsibility to inform those who are making the executive decisions of the consequences of their actions. We cannot expect that everyone is as knowledgeable about what the systems require as we are. Such a move can present tremendous opportunities to eliminate problems or reduce the impact of an unexpected occurrence, but it will do neither if we fail to take advantage of it. The optimal design of the computer facilities can save tremendous costs in down-time, disaster recovery, and inefficient operations. It is our duty both to ourselves and our companies to make these points known. Believe me, this is not a wholly altruistic view. The problems eliminated in the design phase will save you headaches for many years to come. Consequently the ones not eliminated will haunt you, or your successor, for years.

CONCEPTION

THE IMPORTANCE OF PRE - PLANNING

I cannot over emphasize the importance of designing and planning. While it is true that many of the plans you make will either be changed or rejected by yourself or others, it is essential that you take the time to examine all the impacts of a given scenario. One of the first barriers we ran into was the complete lack of understanding of this principle by the majority of the others involved in the project. It was not uncommon to spend hours working out the layout of a particular area only to have some authority figure change everything on a whim. After the first few times this happened, I was ready to hang up my architect's hat and go back to managing Operations. But I learned a valuable lesson.

Don't Throw Anything Away. At least don't throw away any of your early plans or drawings until the final plans have been approved or, better yet, implemented. If possible, use a CAD package for your designs and a word processor for your

reports. This will make it easier to change the drawings when someone else changes your plans. Frequently a concept that had been disapproved early on was resubmitted and approved later. It is also advisable to keep copies of any memos or planning notes that get circulated. They can come back to haunt you later. The basic assumption is that if someone wrote it down, it should be read and either applied or countered. Many of the original concepts will not change. If it is determined that a Haylon system is needed, it doesn't matter if the computer room is moved to the other side of the building or not a Haylon system is still needed.

Establish your basic needs and stick to your guns. By basic needs I mean those that can be justified. I had no trouble defending the need for environmental controls and security doors, but I did have a hard time defending the window for my office (in fact I failed to get one). Those areas that I could quantitatively define, like cooling capacity or amperage, were relatively easy to present and obtain. Even those areas that were somewhat nebulous, but still of a practical nature, like the operators work better without an audience, or some lighting and colors reduce stress and eyestrain were defendable. It was when someone tried for a power play that the justifications broke down. Ask yourself do you need it or just want it? If you think you need it, why? If you really need it is it worth what it will cost? Those are the questions the administration will ask, and if you intend to remain credible you had best have the right answers. Maybe if I had argued the window was a necessary fire escape, yeah ... but what about the Jacuzzi? Stress relief?

ESSENTIAL DETAILS

Rules and regulations. There are considerable restrictions as to how things can and should be done in a variety of instances, and failing to know and follow these restrictions can be extremely costly. For instance, there is considerable discussion as to whether PVC cabling will soon replace asbestos as the number one building bane. Apparently PVC cable releases a toxic gas when burned and if the cables are located in the "return air plenum," that gas can spread throughout the building. Right now the rules say PVC is okay (if you don't mind the thought that you could kill everyone in the building, yourself included, if you have a fire). The cost of "plenum rated" cable is substantially more than PVC, but the costs of pulling out all the cabling and replacing it once construction is complete are astronomical. Not only must you be cognizant of the regulations, but a thorough knowledge of the policies of the vendors is helpful. For instance, at what point does your maintenance contract become void? What are the operating parameters of your equipment and what can you expect if you exceed them? What breaks can you get on insurance if you install a security system? At MWSC we found out many of these details after we started the project (some

after we finished). It would have saved a great many wasted hours if someone had looked into them before hand.

Contingency plans. By contingency, plans I mean simply looking at the plans from alternate possibilities. You know what they say about "the best laid plans." If continuing operations is a priority in your company (and where isn't it?), then train yourself to expect the unexpected. When we get into the "deception" portion of this paper, I will expound on just how unexpected things can get. Every difficulty addressed and planned for is one less that will have to addressed at "the worst possible time." Ask yourself what would we do if...? If the ... is too far fetched you can laugh at the possibility (God knows there will be precious little to laugh at later). If not, you may have saved a great deal of difficulty when it does come up.

Take that ounce of prevention. You know the adage about an ounce of prevention being worth a pound of cure. Well in this case, it is more than applicable. A few hours spent in pre-planning and organization will save, at least, countless hours in implementation and possibly the entire project. Determine the background of the architect and the others responsible for the plans. If they do not have the knowledge required to design a computer facility, then insist that someone who does be involved. If you don't have the knowledge personally either find someone who does and hire them as a consultant, or obtain the necessary knowledge. There are a great many resources available if you decide to take on the job yourself, but remember if you don't know, say "I don't know" and find out. Don't just let it slide. There will be a great many things that are unanticipated, and the more eliminated at this stage the better the chance of surviving with the majority of your mental faculties intact.

DECEPTION

Once you have gone to all the trouble of making and revising countless plans, attending enumerable meetings, and studied every conceivably pertinent piece of material, you will undoubtedly discover that not everyone involved is playing the same game.

THE GAMES PEOPLE PLAY.

The contractors (what you see is not necessarily what you get): In my naivete, I believed that the contractor's job was to follow the plans that had been so painstakingly drawn up. That is not the whole story. The contractor's job is to build as cheap a facsimile of those plans as they can get away with. I know this sounds a bit harsh on contractors, but I've lived through it and it might be a bit lenient. Another mistaken assumption is that they hire trained professionals to do the work. They hire some very knowledgeable professional people, but they also hire some people who give new meaning to the term incompetent. Before you dismiss this as the rambling of

a socially maladjusted computer manager, let me recount some specifics:

On this project, the foreman is a man named John. Now John is pretty much what you expect from a construction worker foreman. "Every morning at the site, you could see him arrive. He stood six foot six and weighed 245." (special thanks to Jimmy Dean for that last) John is soft spoken and easy to get along with as long as you don't ask him to do something he doesn't feel like doing, or expect him to do what he tells you he will do. John and I do not get along all that well, not that we openly fight (I am not stupid), but we seem to have some serious communications problems. You see when I say "that temporary wall must be air tight," I mean sealed so that air does not pass through. John, on the other hand, means that you can only see outside air if you look at the edges. I pointed this misunderstanding out to John and informed him that in it's present state the wall would: allow in enough water to fill the entire sunken floor area if it rained; the dust created from the earth moving and concrete cutting would pass through and destroy my disk drives; if the haylon system were energized, the majority of the gas would be sucked outside and be useless; my air conditioner was not designed to cool the entire city; and an outside wall made of plywood and two by fours with a quarter inch gap around the edges is not the epitome of security. John informed me that the plans did not specifically describe the temporary walls, but if we wanted to put in a "change order" at additional costs it could be arranged. Needless to say, I had not anticipated this contingency, but we reached an equitable arrangement. I put caulking around the edges, asked security to increase patrols in the area, worried a lot, and generally made the best of a bad situation. John went on with his job undisturbed by the terrible things I did to him in my mind.

This was only one incident in a seemingly endless string of confrontations with John and his magical "change orders." Several had to do with specifics that were in the plans, but due to an error or omission on the part of the workers did not show up and had to be rectified. In those cases where I could point to the plans and say "there it is," John was forced to acquiesce, however reluctantly. The important lesson with contractors is that it is your responsibility to know the plans, and insist that they be adhered to. It is also important to insure that the plans cover all temporary or transient stages of construction. If special constraints must be followed around the computer center make sure they are covered in detail in the original plans. Include what conditions will require shutting down the equipment as well, otherwise the assumption is "run it until it drops." Remember the magical change order can fix anything, but the costs are considerable. Also, it is much easier to enforce something that you have in writing as opposed to something that is simply "obvious."

With regards to contractor's hiring professionals: they don't. In fact frequently the contractor hires a sub-contractor who hires whomever they please. At one point I found one worker with a floor tile removed, a connection box opened, and a screwdriver, attempting to "short out that circuit breaker to the wall outlets. Why do you ask?" Never mind the fact that the screwdriver was half the diameter of the wires he was trying to short. Never mind the fact that those wires led to my power conditioner with a one hundred amp circuit breaker. Never mind that if he managed to get a good connection he would succeed in spot welding his screwdriver to the wires and electrocuting himself. What bothered me was that I was the only one present and would be expected to give this imbecile mouth to mouth resuscitation. Certainly some of the workers on these projects are professionals; all I am saying is don't bet your system on it. If the computers are your responsibility, you must be intimately involved in everything that can effect them. During construction there are literally dozens of persons who are unknown seeking access to areas where they would not normally be allowed and maintaining security is extremely difficult, but it is still essential.

The contractor and labors have no vested interest in your operations. It does not bother them if your security, environmental controls, or safety precautions are destroyed. What does bother them is anything that hampers them from doing what they want to do. I came in to work one morning to find an open door into the vault. Normally that would not be of great concern except that this door had not existed the day before and it was open to the outside. When I asked John about it he told me it was necessary so that his workers could "move stuff easier." I quietly relocated the blank payroll and accounts payable checks to a more secure area (the trunk of my car) and retired to my office to think evil thoughts about John.

The other departments: I began this project under the assumption that I would have to contend with a great many "games" from a great many players. I was unaware that several (generally the most obnoxious) of these players would be from MWSC. Some of the deception I encountered was from within my own organization. I should have been ready for it. I had experienced "power plays" before, but not in this particular area. During much of the planning stages I encountered conflicts that really should not have been there. I would design an office for, say, the Operations Coordinator. I would determine what was needed in the way of terminals, P.C.s, bookshelves, file cabinets, desks, chairs, etc. After that I would determine how much floor space was needed, and allocate that much space in the drawings. It seemed like a completely logical plan. Unfortunately a much easier and less controversial method existed. I merely had to determine the pay level of the individual, find someone else at that same level, and measure their office. More then once the principle (pun intended) argument against an office design was that it

was bigger than someone else's. Within certain areas I could be as ludicrous as I wished. Once I mislabeled the electrical connections so that they were cable television connections (it's not as hard as you might think). No one batted an eye. But let some one at level 13 have and office bigger then someone at level 14 and you'd think I was requesting a wet bar for my office (maybe if I said it was a lab bench, but how would I explain the refrigerator?). It should be pointed out that in the vast majority of these cases I managed to successfully defend my position, but only because I was able to present justifications. When I could show that a certain number of square feet was necessary for a definite reason I generally got what I asked for.

At MWSC we have a department called "Physical Plant." For those unfamiliar with the term, that is the department responsible for maintenance and repairs on campus. For them the problems I had were relatively small, because they had similar difficulties with the entire building. During construction there is a nebulous period when nobody knows who is responsible for what (unless it was outlined in the construction plans). If the air conditioner fails and the back up does not work, normally Physical Plant quickly (well quickly for Physical Plant) repairs the offending equipment and all is well with the world. But what if the back-up doesn't work because the construction people have removed the duct work. This situation actually happened at MWSC. After several hours of "finger pointing," it was decided that since John's people where not about to fix it, the responsibility was clearly Physical Plant's (the fact that I told them the system would be shut down in one hour if we didn't get air had no bearing on this decision, or so I am told). The main system was repaired in record time and we did not have to shut down.

I would like to say that was the end of it, but, alas, it wasn't. Being one of those managers who attempts to prevent a future incident by determining the cause of the present one, I inquired as to the cause. The answer made my blood run cool. "It was clearly sabotage" said the repair man. Apparently someone had removed a "wire jumper" between two terminal blocks. This had caused the unit to shut down. It was three thirty on the Friday before the Presidents' Day holiday, and being in charge of security I really did not want to hear "sabotage." But how could I argue? Clearly the system was working now. As I am by nature an extremely dedicated person, one who knows from experience that leaving a loose end dangling over a three day weekend will inevitably produce a noose, I chose to investigate. In a manner that would have made Sherlock Holmes envious, I deduced that it was not sabotage, but rather that imbecile with the screwdriver I mentioned earlier (I knew I'd live to regret telling him about those cables). When he had finally isolated and switched off the appropriate breaker, he had not determined what, if anything, was plugged into the offending wall outlets. So no one knew he had also shut off the condensate

pump on the air conditioning unit. Had the air conditioning unit run over the entire weekend dumping the condensate under the raised floor we would have flooded the area, blown water up through the floor vents (efficiently located beneath the computer equipment), and caused enormous amounts of damage. The "removed jumper" should never have been there in the first place. Our air conditioner repairman had very efficiently bypassed the safety interlock designed to shut the system down if the condensate pump over flowed.

The lesson in the above example is that during the construction nothing is normal. Everyone must expect incidents which are out of the ordinary, and, as managers, we have to be willing to analyze each of those incidents more thoroughly than normally. Had the worst happened, it would have been MWSC that paid for the damages, not to mention suffering the down time. The Physical Plant repair man should have been able to find the real problem just as I was, but he was looking for the most likely problem and stopped when he thought he found it. In addition, he did not have the benefit of first hand dealings with the construction workers. If he had, he might have suspected that someone had tripped the breaker to the pump.

PRECISELY HOW MUCH IS "A LITTLE INCONVENIENCE?"

Prior to the actual construction, I met with the Operations Staff. We discussed some of the potential difficulties we might encounter during the construction, particularly as the related to the work environment. This led to a conversation with John, the first of those, as a matter of fact. This was before I had ascertained the existence of our communication problem. At that time, I asked John what we could expect in the way of changes in our working environment. I explained in some detail how many of us were accustomed to a quiet, stable, smoke free work place, conducive to contemplation and continued operation of delicate equipment. John assured me that there would only be "a little inconvenience."

Environmental stress. On the first day of construction I began to sense that maybe my concept of "a little" and John's were somewhat different. I was unaware that a steel and concrete structure makes an excellent echo chamber ... until they stared using the jack hammers and concrete saws on the outside walls, one of which happens to be the back of my former office. The noise was so intense that even ear plugs failed to dampen it. Phone conversation was impossible, except during their breaks, and coherent thought was something we could only remember. I began to look forward to going home for some peace and quiet. I know most people do that, but most people don't have six kids, a wife, and a dog at home. I invited the Psychology department over as a case study in environmental stress, but they declined saying that they would have to have had tests before to make a comparison, especially considering it was the Computer Center they would be studying.

Our productivity declined to the point of becoming negative. My staff began making excuses to leave the building, and in the winter in Missouri, that is unusual. In retrospect, it would have been better had we set up alternate work spaces in either a trailer or another building. My only consolation in this area was when I considered how difficult it was on the rest of the people in the building. Can you imagine the mental anguish on the librarians?

Once the outside wall was removed, the new foundation laid, and the temporary wall constructed the noise became less intense and less frequent (I won't even discuss those "nail guns" they use without warning). Actually, I am not certain whether that was the case or if it simply paled in light of the other "inconveniences." I am originally from southern California, a native San Diegan, and not particularly well adapted to Missouri cold. As a result I hesitate to complain when my office gets a little cold and simply put on my coat. However, when ice forms on the wall (on the inside), I cannot see my monitor for the fog from my breath, and the wind blowing through the crack in the wall is sufficient to blow a floppy disk off my desk, I begin to feel complaints are in order. At this point, I began looking for some support from the hard working Operations staff. I found them huddled around an HP 3000 looking for a little heat. Physical Plant was sympathetic, but the heating units were maxed out as it was. The construction workers were gone. "You can't expect us to work in this cold."

There were numerous other "inconveniences": power interruptions, dust so thick you could literally not see a foot in front of you (and that was in the computer room), fumes so intense it made everyone dizzy and gave us headaches, at one point a welder managed to set off the haylon system (fortunately an alert operator managed to shut it down before discharge), and countless other incidents. If I had it all to do over again, I would have arranged temporary facilities for the entire construction period. Of course that was not an option for the actual computer room, but even a remote place to escape to would have helped the Operators. The cost of such a facility would have been more than offset by the loss of productivity. We were extremely fortunate in that serious complications did not result (if I could count on that level of luck I would spend my time in Las Vegas at the tables, not a podium).

"When you said you'd give me three weeks notice I thought you meant in advance." Once again my apparent inability to effectively communicate with John created some difficulties (normally I don't have that much difficulty being understood, honest). I actually made that statement to John. Prior to the beginning of construction I discussed, with John, the necessity of keeping me informed as to events that could effect the Computer Center. I even gave him a written document outlining the type of things that we needed to know about. I recall discussing the need for this with an

associate of mine. She felt I was being too explicit, that I might offend John by stating the obvious. To say John was not offended would be an understatement. John was not phased. Oh, he agreed to give me three weeks written notice of anything that would have any effect on us. To date I have received nothing in writing and very little in the way of verbal notice. I will not belabor this point, but it does reemphasize an important consideration. If it is not in the construction plans, it is not required of the contractor. A verbal agreement is not worth the paper..., well you know what I mean. If something is important, make certain it is in the plans and completely understood before construction begins, preferably before hiring the contractor.

COMPLETION

IT'S NOT OVER TILL IT'S OVER.

I am not sure whether it is the little inconveniences, or simple anxiety as the project nears completion, but it seems that the construction work slows as the culmination approaches. It is true that there were times I was tempted to go on a search in the music department for an obese soprano to end the misery (it's not over till the fat lady sings). As the project winds down, more "little things" seem to slip through the cracks. As much as I wanted the project to be completed, I wanted it completed correctly even more. John and I began to play "see who can wear the other one down first." I would notice something that had been forgotten; John would say he'd get right on it; two weeks later I'd notice it still wasn't done; John would say he'd get right on it. I began to make check lists so I wouldn't forget anything. Some of these omissions were pertinent, at least I thought they were, John obviously had a different opinion. For literally months, I tried to impress on John the importance of placing the nozzles on all of the outlets for the haylon system (while the center was operating). His responses ran the gamut from almost believable to totally ludicrous. Once I was told that nozzles were not necessary because the workers had drilled tiny holes in the capped risers. Once the project is accepted, very little can be done to call the workers back and rectify mistakes. Play Santa Claus; "make a list and check it twice." I am not suggesting attempting to do the contractor's job. I know relatively little about most of the work John does, but I do know what is needed in the computer room, and it is my job to insure compliance before the project is accepted.

SOME MISTAKES CANNOT BE LIVED WITH

Many of the omissions we experienced were cosmetic in nature: a ceiling tile was missing, a piece of molding forgotten, the cover on an electrical outlet damaged. These small items could be "lived with," or even personally addressed. On several occasions, it was much easier to do the job myself than to add a minor item to the list and appear to

be nitpicking. Some mistakes, however, cannot be tolerated. Either they are too large or they will result in too great of a consequence. An example of one such mistake occurred with the sub-contractor responsible for providing the phone and data outlets throughout the LRC. We had already resorted to the magic change order to correct an oversight in the original plans ("Oh! You wanted to have the data connections in the programmers offices during construction."), so we were cautious with this particular sub-contractor. The plans called for a detailed diagram depicting the location of each of the cable runs. A diagram was provided (some people have a unique concept of the term "detailed"), and after some translation work, we could read it. The problem was it was not accurate. "Oh, that doesn't matter just tone it out." I explained, as calmly as possible, that I had no intention of "toning out" 1,600 twisted pair and correcting the mistakes, but that I did intend to insure it was done properly prior to final acceptance. Even this turned out to be a compromise, because we had to redo the corrected list. The point is: If you don't insure the job is completed correctly, chances are it isn't.

WHEN YOU'RE UP TO YOUR REAR IN ALLIGATORS .....

As the project came to a close, John and I were not the only ones anxious to see it finished. Deadlines were rapidly approaching and pressure was building to "sign it off and be done with it." Although the temptation is great, remember your objectives. At the beginning of this paper, I expounded on the need of pre-planning and attention to detail. I firmly believe that it is during the last few weeks that those original concepts are lost. Everyone is more than ready to see the job completed and in walks someone with a written list of discrepancies. I was not very popular at that particular meeting, but I got those nozzles. It makes no since to start a project like this and not see it through. Realize, from the onset, that eventually it will come to this: if you are not willing to fight for what you need, you are not likely to get it. I realize I have been a little hard on John and his workers, and some of it might have been unwarranted. John, like most contractors, does not have a great deal of experience building computer facilities (ours made one in a row). He cannot be expected to know what needs to be done. I am certain John did not appreciate my dogmatic insistence on what he felt were trivial matters, but I was responsible for the functionality of that facility, and I refused to allow inadequacies simply because I was to lazy to defend my positions. Forewarned is Forearmed. Realize from the onset that you will have to fight for everything you need and be ready to defend your position. If the project goes off without a hitch, fantastic; if problems arise, you're prepared.

CONCLUSIONS

Thanks to an outstanding effort by everyone involved (yes, even John) the project is complete. The computer room is an environmentally sealed steel and concrete 32 by 44 foot room with an attached vault, storage area and loading dock. The operators have a comfortable room where they can work without an audience. The entire building has modular data jacks which connect to a data distribution switch (acquiring that while all this was going on is another story) through clearly labeled punch down blocks. Our offices are sufficient for our needs with some room for growth. The student lab and classroom areas are properly lighted and equipped with sound dampening wall covers to reduce noise. They have special air conditioning units to compensate for the heat generated by the terminals and micros in use. The computer room is equipped to allow it to become the telecommunications center in the near future with space allocated for a voice PBX, video distribution equipment, and all necessary wiring (I didn't want to go through this again).

For the most part, everyone has recovered from the experience nicely. In fact, some of us are getting a little restless. I almost miss the anticipation of wondering what new challenge will await me when I get to work. My hearing is almost back to normal. The experience wasn't all that bad, and look at what we accomplished. Now that I think about it the Math Science building could stand rewiring and the Admin. building...now where did I put that CAD package?

The Small Company Challenge: Strategies for Managing Software Development with Limited Resources

Cindy Curtis and Cheryl Morse
INLEX, Inc.
P.O. Box 1349
Monterey, CA. 93942

Companies, large and small, share the problem of having the demands of the users become too great for the current resources. A large company, padded by the luxury of size, has the ability to endure situations that might otherwise destroy a smaller business. Problems revolving around finances, human resources, and productivity are magnified, and often fatal, for the smaller institution. And yet, many small companies thrive even when they are under the reins of limited resources. Our focus is to discover how management can make the difference.

Much of our discussion could be labelled "old hat," for surely, many of these techniques have been known for several years. Our intent is to illustrate, through specific examples, how implementation of some of the often overlooked techniques could bring about small, but effective, improvements in productivity.

While the central theme of this paper concentrates on the problems encountered when managing software development, many of these problems can be applied to other environments as well. Some of the solutions that are presented include:

- Develop a goal for your department and aid your company to do the same. This will ensure that all efforts are in the same direction.
- Appreciate each employee for the benefit of their own contributions. Strive to make all employees responsible for the success of the company and share in its achievement.
- Build a unified base of employees that work together as a unit to ensure the success of the team. Endeavor to keep the team by providing an enriching work environment.
- Optimize your hardware budget by considering some unorthodox strategies including equipment trading, maintenance contracts, and the peripheral market.
- Make the most of the software developed in-house. Use the construction of primitives, standard include files, and filenaming conventions to allow for easier reuse of code and reduced maintenance time.

- Work with the user to provide specifications to ensure the most accurate feedback. Correct design flaws as early as possible. Don't overlook the importance of speed, accuracy, and appearance to the overall usefulness of the program.

- Begin documentation from the earliest stages of system design and include documentation writers in all system discussions. Use selection of filenames and version numbers to provide additional documentation within the code itself.

- Coordinate the release of new program versions on a predetermined schedule and document, document, document.

ESTABLISH A COMMON GOAL

Every company needs to establish a common goal for its employees. The company will only meet its desired objective in the most efficient manner by having the same focus or direction from each employee. Company goals can be as diverse as "Quality software," "Number one in sales," or "Leading in innovation." For some small companies, "Survival" is enough. The selected goal may be grand or simple, but above all, it must be consistent throughout the corporation.

Without a common goal, there could be a tendency for each department to set and follow individual goals that lead to counter productive activities between departments. For example, a sales department that focuses all energies into increasing market share may force the development department to perform tasks that inhibit its goal of quality software. Conversely, a development department that insists on 100% bug free code may force marketing personnel to continually fall back on delivery dates and schedules.

Consider the impact if the employees of a major automobile manufacturer were directed to meet the goal "Quality is job one," with the exception of the paint department, whose directive was "Always true blue." The result might be automobiles with inferior blue paint jobs, that would otherwise be of highest standards. The entire company suffers because of the wayward directive from just a single department. Naturally, this occurrence is highly unlikely, but illustrates the impact of different directives and how it can be devastating to a small company.

Establishing a single common goal for the entire company may prove extremely difficult. Attempts to define the primary objective may yield false aspirations, and not reflect true direction. The company goal must be viewed as a plausible and obtainable objective for all departments. It would be

foolish to expect the customer support staff to become as enthusiastic as the marketing staff over a company objective such as "Market domination in New Zealand."

If the company as a whole cannot agree on a single goal, should individual departments bother to define one? Absolutely. The company will still benefit from the organization of departments that have clearly defined their desired aim. In the case of conflicting objectives between a customer support team, whose goal is "Customer first," and a development team, whose motto claims, "Quality product," each department is able to identify the other department's strategies and focus, and can use this knowledge to foster better communication and cooperation.

Who should choose the company goal? Most likely the task will fall to the company executives from the highest levels. Unfortunately, these may be the employees that are least in touch with the daily workings of the company. Instead of seeking a practical target, they may focus on a objective that is not entirely feasible. To define a course that highlights "Quality" is admirable, but if the actual focus of the company is entirely on "Staying out of the red," the stated goal will only serve as an anchor to the employees that attempt to meet it. It is better to select a workable and realistic goal than one that merely sounds good.

ORGANIZE THE APPROACH

The next major step in resolving the resource crunch is in recognizing it. As a manager, it is easy to spot the areas that are always behind schedule or problematic. But look beyond specific departments, at the company as a whole. Recognize that some of the problems may be accentuated, or even caused, by other departments. Cleaning up the activities in one area may yield less than satisfactory results if the source of its problem lies elsewhere.

Too often managers deal directly with the small day to day issues that are currently pressing, without making the time to address the larger overall issues. In a typical "forest for the trees" vision, the manager may not see how the productivity chain is clogged by inefficient or unnecessary activities and procedures.

Identify a list of problem areas that appear to be good candidates for improvement. Expect the greatest change in areas that are comprised of fewer employees and have fewest interactions from outside departments.

When considering potential solutions, do not discard changes that produce even a 3 to 4 percent increase in productivity, for a few small changes at the top levels can gain momentum and produce massive changes at lower levels.

INVOLVE ALL EMPLOYEES

Once a goal has been established for your company or department, it is imperative that all employees participate in its achievement. Stress the company formation as a whole, rather than as a conglomerate of departments, or individuals. Remind employees of the close coupling between departments that serves as the skeletal support, without which no company activities would succeed.

A company, similar to a chain, is strengthened by its strongest links, as easily as it is diminished by its weakest. A manager must learn to motivate each employee to provide their best effort. To do so, the manager must make the job "personal" to each employee. That is, make every worker responsible for the end result. Employees perform better if they believe their personal reputation is "on the line."

Each employee must be directed to evaluate all commitments against the corporate objective. It is possible you will spot several instances where your company could falter if it proceeds to operate without the selected goal in mind. For instance, is it feasible for a company, whose primary goal is "Survival," to adhere to outrageously brief production schedules, resulting in penalty charges and bad will with customers? Over time, the negative forces will build, eventually resulting in diminished sales; directly countering the prime objective.

Employees become easily discouraged if they believe their contributions are futile or not recognized. Avoid identifying key people as the sole initiators of successes. Instead, let all employees share in the successes of the company equally. The end result is a unified base of employees working together as a unit for greater productivity.

Be prudent when providing bonuses and rewards. Establish company wide incentive programs that transcend department boundaries or do not provide them at all. An incentive program that allows only select employees to benefit causes the alienation of co-workers and establishes an arbitrary boundary between employees. You would no sooner set up a distinction between employees by age or shoe size - why encourage other artificial boundaries? Perhaps, a better distinction would be by project, or task, whereas the entire team of players is rewarded at the culmination of the accomplishment.

Reconsider incentive programs that are already in place. Though rarely the sole force behind any sale, the members of the sales department, through commissions and bonuses, are

often the only employees that directly benefit from this accomplishment. Contributing departments, no less a major player in the achievement, are often overlooked, and remain without recognition. Make a point of passing the praise to every employee for each major step towards the goal.

Another example of the potential damage to employee morale involves something as trivial as parking. Having a section of parking spaces reserved for high level staff may be considered unfair by those employees that are of lower company stature. Spaces reserved for company executives or other personnel often away from the office will remain empty while other employees must park at the back of the lot. Remove these unnecessary outward appearances of inequality between staff, and you will discover that employees that feel more like the members of a complete team.

When all employees feel that they benefit from their contributions in an equal manner, each will be more likely to work to their full potential. Ask for total commitment from each employee, and strive to make full utilization of human resources.

BUILD THE TEAM

A good team is necessary for the smooth implementation of any project. Individually, an employee may perform in a competent manner, but unless he can also contribute to the department as a team member, his full potential will remain limited. A premiere programmer may produce code that is intensely complex and supremely efficient, but unless the code follows conventional standards and is easy to maintain, the usefulness of the final product is drastically reduced. When the programmer leaves the company, the program may be rendered useless because it cannot be understood enough to be modified.

In a software development group, following a team approach includes showing concern for future code modification by others, courtesy with respect to system use, willingness to follow group standards, and arriving prepared and prompt for staff meetings. A programmer that always runs jobs in the highest queue with little concern for the jobs of others is not a team player.

A good team is far greater than just a collection of employees. A manager needs to hire persons with the entire team construction in mind. He must look for a group of people that will work well together for a long term. Too often, managers emphasize experience and skill more than personality. A team member that is unable to work within the team because of personal conflicts cannot contribute

their best effort. The most experienced and knowledgeable candidate is not a good match if he will not be able to work within the team.

Hire persons that are willing to perform more than a narrow job description: i.e. "a person of many hats". Encourage people to gain experience in all areas, including documentation, training, MIS, development, support, and marketing. Bring programmers to the sales conferences for exposure to the difficulties marketing personnel face. Let customer service participate in development design sessions. Bring trainers to user group meetings to discover the "true" nature of real life problems. This cross-training of employees will yield a team that is empathetic and more responsive to the needs of other departments, even when performing their own duties.

Hiring personnel with the right match is more important in a small company because there is no easy way to absorb a bad choice. In a large company, the unproductive employee can more easily be sent on an endless transfer to other groups, eventually dumped at the department most known for stagnation. Small companies have no place available for "dead wood"; any employee that does not fully contribute must be trimmed from the tree.

Like cars on a highway, traffic is impeded by drivers that travel too fast, or too slow. The optimum movement requires a consistent and similar speed from all automobiles. In relation, department productivity may be hampered by employees that work "out of range" with their co-workers. Take steps to keep the effort of staff members at a consistent level. Refrain from hiring individuals that are markedly different in their levels of skill and speed. Though difficult to believe, an over productive employee may have as much a negative effective as an unproductive one. Co-workers may give up in frustration, feeling unable to pull their own weight. More importantly, do not let one member of the staff contribute significantly less than the others. Resentment will surface, and the team may suffer reduced productivity as other members reduce their work to the lower levels of the privileged one.

Reduce the opportunity for conflict; time is not available for the smoothing of ruffled feathers. Do not lay blame or point fingers when problems surface, as this results in defensive posturing from the persons involved. Attempt instead to utilize a "joint" approach to problem solving. Let the employees know you will back them up, and not side against them. Make every person in the team equally responsible for any faults.

Keep communication lines open to other departments. Do not hide problems; share them. Let difficulties be known as soon as they are recognized. Encourage a weekly progress review meeting and document the progress from each member of the team in rough form. Once a month, produce a formal version of the progress reports to inform other departments of the work completed along with any current and future commitments. If the reports are received in an impartial manner, the status of projects will be conveyed naturally and without surprise. Emphasize the positive accomplishments that have accrued during the previous month, and award responsibility to the whole group.

KEEP THE TEAM

Once established, a productive and compatible team is an asset no company can afford to lose. Small companies have the advantage in that they can more easily restructure their policies to accommodate unique employee situations. Take advantage of this distinction in the areas that will best benefit your department.

Training costs are especially high for small companies since funds are rarely available for training classes. New and current employees must be self trained or require available time for instruction from existing staff, resulting in diminished productivity for the duration of the initial period. It is important for the company to factor in these potential costs when evaluating the requirements to keep the team they have.

Some managers believe that accommodating the personal situations of their employees will yield a more productive employee. For example, consider flexible working hours. Let your staff judge when they might be most productive and work with them on their schedule. Some programmers may prefer to work during the hours when the system is the least busy. Other employees, seeking to advance their education, may wish to schedule their hours around class times.

Attendance to events, such as a user group meeting, conference, or class, is often seen by employees as a privilege. Work within this view and require the fortunate employee to present a summary of the lessons learned, or the activities that occurred. Not only do your other employees benefit from this knowledge, but it removes the need to send every employee. Establish the privilege on a round robin basis encompassing every employee in the department, providing equal opportunity for all to participate.

If all employees are introduced to a compatible level of knowledge, the impact of one member leaving the company will be much less. An unfortunate, but common occurrence, is an

employee that receives specialized or advanced training (such as System Internals classes) at the company's expense, then departs with this knowledge. If employees were required to share the knowledge they acquire at company provided events, the pain of desertion would not be as severe.

Small companies are often unable to pay employees the same salaries as larger companies. When pay scales are not high and benefits are limited, find other perks to reward and encourage; such as optional holidays, accrual of sick leave and compensation for overtime. Do not overlook seemingly minor benefits like payment of dues to professional organizations, or of subscriptions to professional journals. Give your employees the impression they are valued beyond the level of their job.

CULTIVATE COOPERATIVE ATMOSPHERE

The work environment is one area where the small company may be at a disadvantage to the larger company. Money is simply not available for top notch furniture and accessories. The means to reverse this discrepancy is to be more flexible with the work environment.

Keep the company image informal. Let employees dress as they wish to reduce the need to "finance" a fancy wardrobe. Many employees work better when they are casually attired, including the management. Barriers are more easily crossed when the employees feel they can relate to each other on multiple levels. Dressing to minimize the visual appearance of levels can encourage productive interaction between management and employees.

Promote a high degree of communication to employees of all ranks. Hold periodic, but frequent, staff meetings to divulge company choices or to poll employees about pending decisions. Let employees feel as if their opinions count. Staff meetings are, at times, the sole opportunity for all employees to gather. Take this time to do more than a presentation of calendar events. Seek to divulge major company choices or poll employees about pending decisions.

Lessen the normal tensions that may arise from meetings held in the company board room by scheduling meetings at a neutral location, such as off site (at the local park if necessary). The neutrality will encourage employees to speak that might otherwise be too intimidated to do so.

We too often hear the excuse, "I was never informed" in response to queries concerning missed deadlines or misinterpreted specifications. Insist that a subordinate communicate to upper level employees via his immediate

supervisor. Small companies, intent on retaining an "open door policy" approach to management, suffer the consequences of middle management bypass in poor vertical communication. Close the gap, not the door. Encourage communication to travel along predetermined routes.

Increase the communication between departments by establishing a series of informal bring your own lunch meetings where each employee is spotlighted in turn to discuss topics related to his position at the company. For example, remove the mystery involving database access after a system crash by having the database administrator explain database recovery procedures. Employees as a whole will respect the work performed by their peers if they are given the opportunity to understand what that work entails.

One of the biggest problems facing small companies is an expansive philosophy that encourages over commitments. Have each employee maintain a list of every project, large or small, that is currently under his domain. Determine who will make the primary commitments for each employee, whether it is the employee, the employee's immediate supervisor, or management. Have all other persons check with this person before making a new commitment or altering priorities on existing ones.

Get users involved in setting priorities. Show them all items on a task level along with an estimate of the time and resources required for each task. Let them indicate the top five, or ten tasks. Concentrate most of your efforts on these assignments. Be responsive to the appearance of common user anxieties by providing periodic sessions for users and others to meet and talk about the direction of projects, gripes and potential problems.

Dispel the possibility of undeserved criticism by documenting major decisions during the life of a project. Each team member should keep a log of phone conversations with other team members and users. Major commitments made between group members and different groups should always be documented. Consider sending out a memo to review commitments made as a follow up to important conversations, whether made on the phone or in person. An electronic mail system is extremely useful for this undertaking.

MAXIMIZE EXISTING HARDWARE

A common dilemma faced by managers is the scarcity of available funds to purchase needed hardware. There are techniques available to stretch limited financial resources to their furthest potential. To start, determine how current hardware is accommodating your needs. If performance is sluggish, or physical limits tight, look to

optimize the equipment you already possess before you purchase more. An inventory of the existing equipment is the logical first step. It will let you know exactly what you have and where it is located, as well as allow for improved tracking in the future.

Attach a sticker on the back of each component identifying data such as the date of acquisition, and the dates and types of maintenance. Assign and include a unique equipment number. Copy this information to your inventory list, along with additional notes including the user, building and location, and the overall condition of the piece. Highlight the items that are in especially bad condition and will require replacement in the short term.

Placing maintenance information directly on the equipment sticker makes it easier to track the history of repairs. This method will assist in identifying when the same piece suffers repeated failures, or if multiple pieces require a similar repair. When money is finally available to purchase new equipment, the maintenance "histories" will point to the obvious choices and avoidances.

Eliminate the guesswork on loose cables by identifying pin connections. The next time a cable is needed, it can be immediately utilized without the usual trial and error. Take a little extra time during the inventory to label all port connections at both ends. If you are feeling ambitious, sketch a layout of the equipment showing location of each port (your nighttime operator will appreciate you for this when he has to abort unidentified users who fail to log off).

Seek out methods to maximize your existing hardware. Reexamine solutions that you may have discarded in the past because you felt they were unfeasible. In really tight situations, they could provide relief. For example, consider sharing terminals or personal computers for employees that use the devices only a portion of the day. Staggering their work hours could relieve some of the potential conflicts in use.

Some shuffling of equipment between employees is another quick fix. The personal computer on the receptionist's desk may contain the hard disc drive required by the accounting clerk. Perhaps moving employees to adjoining offices will allow a dedicated printer to be shared among many users, utilizing something as simple as a multiple switch.

If employees are willing, let them bring in their own personal computers and printers for office use, but be sure to label them as such. If the equipment is in good shape,

the company may later opt to purchase the employee's equipment. A price could be negotiated that is very reasonable to both parties.

If your disc space is squeezed tight, impose a policy to archive files that have not been accessed for a lengthy period. Retain a hard copy listing of files for inquiry purposes. For safety's sake, always make two copies, and store one off site.

Begin to track the physical distribution of your equipment on a regular basis, annually at minimum. Sloppy hardware management encourages theft and loss. A lot of equipment dollars may be walking out the door with your employees. Insist on tracking items that are removed for off-site use. In the event the recipient should leave the company, it could guarantee the item returns before his final day.

When time comes to purchase additional hardware, invest some time in comparison shopping. Do not assume that the original equipment supplier is the sole source. Investigate the "pre-owned" market. Like most household appliances, hardware depreciates greatly in value once it is installed. Let some other company absorb the immediate loss of value by purchasing equipment that is at least two years old. It will still be new enough to be current, but has already endured the worst of its value loss.

In the peripheral market, a bargain is a draw for all of us, yet we realize that a cheap piece of hardware may cost more in the long run due to increased maintenance costs. But do consider purchasing the less expensive "gray market" goods in two instances - if they are needed for just a short while, or if they are expected to receive an incredibly heavy amount of use. For instance, a traveling salesman may choose to purchase a single expensive sedan for $35,000. The machine would endure extremely hard use, and require extensive maintenance to prevent failures, and still, the car would be subject to destruction by overuse. The salesman could be better off if he were to invest an equal amount of money into "a six-pack of Yugos," discarding them when they burn out. Used peripherals are about as market worthy as week old bread. Count on the total devaluation of any peripheral to occur no matter what its original cost.

REDUCE HARDWARE SUPPORT COSTS

Have you considered that the maintenance contract on your older model CPU may be costing you more per year than to purchase an identical system from remarketed equipment? Be aware that the company that sold you the hardware will not tell you when it is unfeasible to continue supporting it. If the cost of the maintenance contract is equal to one

third the cost to replace the hardware, then consider halting the service. Siphon the money saved into your new equipment budget.

If eliminating the contract is unfeasible, consider reducing the service response time for a significant reduction in cost. If immediate servicing is required, it is probably cheaper to pay for a third party maintenance firm on a time and materials basis.

Even if you retain your service contract, investigate the other options available before you commit to repairs. A system manager was not informed by a major computer vendor that for the cost of $6000.00 to repair her old disc drive she could have, more easily, purchased an identical used one. Instead, she suffered an extended period of down time while the company sought a replacement part for the discontinued drive. Her system would have been "repaired" more quickly, and cheaply, if she had simply thrown the malfunctioning drive away. But, because she had retained the service contract, she felt obligated to use it.

OPTIMIZE THE USE OF IN-HOUSE SOFTWARE

It is difficult to get money approved for the intangible benefit of software. The inability to illustrate results until after purchase and installation leaves many potential productivity tools unreasonable for the small company. There are ways, however, to achieve improvements in productivity without the expense of purchased software tools. It all leans on organization.

Many basic systems consist of IMAGE databases and KSAM files. The activities to put, pull and locate data within these files are nearly identical in execution. Reduce the effort and duplication of routines that access these files by constructing primitives for activities against every database and file set used within company software. In addition, build standard include files to define the constants, type specifications, and variables used with each file set. Be careful when including more than one database or file definition within a single include file, it may force you to include more declarations than you intend to use.

The primitives must remain consistent in activity and style to eliminate the uncertainty of processing. For example, do not have one of the database write routines perform an internal lock and unlock if all others do not. Once the original primitives are established, replicating the primitives to address other databases and files is as simple as changing the names and lengths of files. Programmers can rely on the primitives as "tested" components in new

programs. As existing source is modified with enhancements or fixes, replace the old database access routines with calls to the new common constructs.

Keep the source for the routines and data definitions separated and named by the originating database or file, (i.e. CUSTOMER.DATABASE would be accessed by routines located in the source file CUSTOMER.DBACCESS, defined by variables located in CUSTOMER.DEFINES). Enforce naming strategies that allow programmers to quickly identify which file sets have available primitives for inclusion in new code.

Expand the "primitive" philosophy to include all code generated by the company programmers. Insist that programmers write modules in tight formation that promote sharing and reuse. Put the most often used routines into separate source files that can be included in any program. Some examples of code that would qualify are date conversion routines and the printing of common report headings. The intention is to ultimately provide a "pool" of source modules from which all new programs are pieced.

Make use of software that is already available through the contributed library or the software that came with your system but has never been used. This may include debuggers, documentation aids, resource monitors, and more. Assign the investigation of the tool to one of the staff, who will then be able to instruct other members of the new capability.

Standardize company wide use of software for particular functions, (i.e. allow use of one editor, one word processor, and one electronic mail package). Companies that allow every employee to select and use different software for similar functions are unwittingly perpetuating the limits to which machines and data can be shared.

For personnel other than programmers (who are always a bit picky), standardize the directory structure within their personal computers. When the software and the directories are the same on all personal computers, employees can transfer from station to station without guessing at existence or location of software. Simplify the installation of new machines by creating a "master" installation kit that includes all defacto software for personal computers (be sure to check for copyright violations, of course).

UNDERSTAND USER NEEDS

Program rewrites are probably the most costly event in software development. Unfortunately, small companies rarely have enough staff and time to perform complete

specifications for every project. The good news is, full specifications are not always needed. Of course, we are all aware of the highly visible and expensive project that, even with extensive specifications, was rejected by the end user. Prevent the opportunity of failure by fully understanding the user requirements prior to releasing any software into production.

One of the hardest occurrences for programmers to accept is the fact that programs must change. It is difficult to acknowledge that design specifications evolve over time as user needs are revised. Be prepared, and anticipate change by reducing the need for the ultra specified system design, replacing it with a more adaptable form of dynamic design.

One inroad to dynamic design specifications is the white board. Consider placing white boards in every office and conference room. White board specifications encourage quick thinking and interaction between participants. They facilitate commentary that hard copy printed specifications do not. A massive document, impressive by size and effort alone, is enough to dissuade revision suggestions from staff, whereas a white board does not present such a rigid design limitation.

Retain a folder for each major project into which you place all scraps and sketches of system design. Any person reviewing the folder is able to travel through the progression of the project as it changed from conception to completion.

Understand that users are not able to easily visualize activities performed by computers, especially when the activity involves screen manipulation. Words alone may not accurately convey the anticipated outcome resulting from the pressing of a function key. Work within this veil of communication by providing specifications that users can "touch and feel" so they can give you more accurate feedback. Step by step, the use of screen visuals and production prototypes that mimic the final system functions provides a means to create projects with only a limited use of specifications. Do not worry that the "test" system will evolve into the final product - that is the desired result.

Implement quality assurance all along the development path to provide checkpoints to spot potential problems or correct design flaws at their earliest moment. Include the end user in the review to be sure the final "customer" is satisfied. Too many projects have been completely finished, only to meet with disappointment from the people who have to work with them. Avoid costly mistakes by knowing the importance of end user approval.

Failure to meet real life levels of use results in the disuse of many systems. Part of quality assurance includes checking that the programs will meet not only the specifications but also the expectations of speed, activity, and appearance. Realize that QA and testing cannot always be performed completely before production release. Enlist the aid of all personnel involved in the project (marketing, technical writer, etc.) to watch for inconsistencies and bugs. Consider a reward system to encourage the reporting of bugs, especially before the product is released.

For cases when software cannot be generated in-house, consider contracting out specific software requirements. Under special circumstances this may be more cost effective. Be careful to draft tight specifications and monitor the development closely over the life of the project.

DOCUMENT AS PART OF DESIGN

Product documentation is the most likely candidate for dismissal in a project pressed for completion. Often slated as the last phase, it is frequently performed in a hurried and sloppy manner, or more often not performed at all.

The documentation which is produced is at best inaccurate and incomplete, being based on misconceptions about system performance and abilities. While writing documentation, writers are repeatedly viewing the systems for the first time, and are forced to draw conclusions about functional logic based on appearances of activities, not true actions.

The existence of internal and external documentation for a system can mean the difference between maintainability and frustration for subsequent users and programmers.

Encourage that the creation of documentation occur at all phases of the design and development of a project, not entirely at the end, by including a documentation representative in development meetings. The writers will gain a deeper understanding of the system. They may begin to sketch rough outlines of early draft documents from the start of design decisions, generating a form of specification documentation from the start. This also eliminates the need to instruct them on the system at a later date. Users should be allowed to review the early drafts to provide feedback about direction and functionality of the project design. Make no attempt to finalize the content of these early outlines. Save the last checking for the final release. Major flaw corrections or functional changes can be incorporated as the development continues.

Force internal program documentation to include, at a minimum, an explanation of how to create and run the program. This will ensure its usefulness to other

programmers. Include a maintenance history showing when the source was changed, why it was changed, who changed it, and the new version number of the program. Document each routine with a preamble that explains the purpose of the routine and requirements for each parameter. If ambitious, list each global used in the routine.

Be descriptive when assigning release and program version numbers. They should tell you more than just the number of times the program has changed. Use a system that identifies a date, as well as a level of change (ie: 891231.1.a). If your programs compile multiple source files, make sure you track a version for every source file. It is imperative that any changes to source code result in an increase in the version. Keep program versions separate from source file versions - or link them together in a meaningful manner.

Provide another form of internal program documentation through smart use of file and group names. Choose file names that are demonstrative of the primary purpose of the file or activity. When multiple files are involved, prefix the names with a brief, but identifiable mnemonic. Instead of a catchall group name, like PUB or JOBS, differentiate the groups into more detailed functions such as RUNJOBS, BUILDJOBS, and COMPILES. Relate the source file name all the way through associated files, such as PAYMENT.SOURCE compiles to PAYMENT.PROGRAM, runs via PAYMENT.RUNJOBS, and uses parameters located in PAYMENT.PARAMS.

SIMPLIFY RELEASE PROCEDURES

As often as changes are required to programs, new versions must be released into production. Along with each release is a complicated series of checks and balances that occur to document the event - which source was replaced, what new version numbers are used, which programmers performed the changes, dates of release and so forth.

Since the complexity of the release procedure cannot adequately be reduced, the best attack is to reduce its occurrence. If possible, limit new releases to once or twice a year. Set a fixed date and adhere to it tightly. Enhancements, more so than fixes, can be delayed in this manner.

Enforce a waiting period for changes to released programs. Insist that users "live" with changes for a period of time before further changes are implemented. This may serve to cut back on hasty change requests, and requests that are not fully thought out. Consider developing forms for reporting software problems and other forms for reporting suggestions and enhancements. Forms should be on line for immediate access.

Prior to each release, complete any unfinished enhancements or corrections and then implement a software freeze. Restrict the initiation of additional changes until after the release is passed to production mode. At this point when the software is in a "known state" of relative stability, save a copy all files necessary for the system so that this start can be returned to at a future date if needed.

Generate a release document highlighting the fixes and corrections, giving details that can be understood by both users and technical staff. Include in each description the identification of source code, program file, and other files involved in the new software. Don't forget to stipulate dates and version numbers. A summarized version of this document may be appropriate for inclusion along with the new release when it is sent to the users.

Establish patch and release history documents to control all changes that occur after a product is released.

If programs are used at more than one location, maintain a site history of all of the programs delivered and installed. Time is wasted when support cannot trace the exact version of code installed at a site.

CONCLUSION

Managing software development is a complex undertaking. This is especially true in a small company where resources are limited. Funds are not available to provide for enough staff, software tools, and equipment, to perform up to the expectations of management. All is not lost. A few simple techniques will result in increased employee productivity: develop common goals, appreciate each employee, build a team, optimize hardware and software, work with users and document extensively. These techniques are easy to incorporate and require no outlay of cash or resources. Remember, small companies are flexible and can more easily change policies and direction than large companies. Small is beautiful.

AUTOMATED OPERATIONS

Robert Forbes
Carolian Systems International Inc.
3397 American Drive, #5
Mississauga, Ontario
L4V 1T8 CANADA

This paper discusses centralization and automation in a multi-HP3000 environment. It offers suggestions on how to plan for and introduce network-wide operations control, thereby making large shops more manageable and efficient. This paper deals with centralization and automation as two distinct issues, considering first where centralization is required, and only then implementing the automation process.

INTRODUCTION: THE OPERATIONS OPTIMIZATION PROJECT

In recent years, multi-system installations have become more common in the HP3000 community. While the advantages of both distributed and local network systems are often obvious, the challenges associated with their management as one coherent unit are all too often underestimated. Since each machine is typically monitored and controlled through its independent console, an overall perspective of network-wide operations is difficult, if not impossible, to obtain. Yet sensitivity to global issues and centralized decision-making are essential to the efficient management of large scale integrated systems.

The problem of centralized control over multiple 3000s is vastly complicated when a central site has to support remote sites. Often a lack of adequate operations staff limits control over production and severely impairs the flow of system status information back to head office. Similarly, the sheer size and complexity of local multi-machine installations can also render traditional operations management techniques ineffective.

The possibilities for streamlining networked operations do exist in an HP3000 environment, and provide several advantages over traditional operations procedures. A well-planned centralization project will allow you to develop the tools with which you can centralize your information and which will give your existing operations staff greater control over distributed systems from one location. All of this can be achieved through a program which takes into account important hardware, software and network issues, specifically in the areas of automation, integration and compatibility.

The resulting centralization of information not only makes your operations more efficient; it also provides management with procedures for production. Such policies inevitably lead to improvements in overall system throughput and higher system utilization. Moreover, continually reviewing operations management systems guarantees the optimal use of existing resources as installations grow.

What is needed, and sometimes missing, is a firm commitment to an ongoing, long-term operations optimization project. Although this will divert resources away from satisfying some demands of the data processing department in the short term, failure to review procedures may significantly erode its operating efficiency and ability to service the company in the long term. Since the results of your labour will be a long term solution, you can implement distinct phases without incurring prohibitive expenses, while at the same time remaining confident of a proper design and

direction. In the long run, it will be well worth the effort. In tackling the issue of centralizing and automating your DP shop, it is important to understand fully how your operations work and, more importantly, how you would like your shop to work. Therefore, I would like to review some of the basic concepts about the operations environment before continuing.

Today, HP3000s are controlled by an operator through the systems' local consoles. This system console is the lifeline to the HP3000, since all operations status messages pass through it. It is a hub where all problems can be defined in two categories: regular, day-to-day operations messages (such as tape requests), and messages that report an error condition to which someone should be alerted.

In this scenario, it is assumed that trained operators are constantly monitoring the console screen so that they can respond to the requirements of the message immediately. This also assumes that the operators are trained in a variety of areas, so that they can handle job management issues, system security policies, internal system management, data comm, and so on. In short, good operators have to specialize in many different areas -- they must be a jack of all trades. Traditionally, the operators must also contend with a variety of messages which are irrelevant to their primary tasks.

For example, if the operator is mainly concerned with executing and monitoring an installation's production schedule, then that person will only want to view job completion and error messages from among all activity passing through the console. He or she would not be interested in logon messages, HPDesk messages, data comm messages and so on. The result is an operator who must act as a human filter. As the site becomes more active or complex, the operator's job becomes increasingly more difficult as critical information scrolls off a busy console and disappears. Then the operator is faced with paging through hard-copy $STDLISTs.

Now imagine this task in a multi-machine environment. Busy shops might need an operator per system in order to keep operations running smoothly. Such a setup, however, makes centralization difficult since each console is physically separate from the next. Therefore, messages are received by operators based on the physical, rather than on the logical structure of the operation. Because messages coming from multiple machines cannot be grouped logically, you must have several operators monitoring separate machines, looking for and responding to only those messages which are relevant to them. This often results in wasted time for a number of operators since they are only concerned with a small percentage of the total messages. In addition, this can lead to a duplication of tasks since two operators may be performing the exact same function on totally separate machines. If, however, messages could be grouped logically, an operator could receive all messages relevant to his or her task from all the machines in the network. With such a setup in place, the operators' tasks could easily be organized according to function rather than to the physical location of the machine, and such centralization of resources and information would thereby result in the optimization of your operations practices.

THE FIRST STEP TO IMPROVEMENT

Throughout this discussion I will be making general recommendations which, when grouped together, provide the basis for a long-term operations review plan. The first of these is as fundamental as it is self-evident: Commit resources to examining existing operations practises through a long-term operations optimization program. This must

be an on-going project so that your operations optimization can accommodate changes in demands.

As I alluded to earlier, the most substantial obstacle this kind of project must contend with is the difficult task of providing operations staff with both the information and control required to efficiently oversee multiple systems. In order to overcome these obstacles, it is crucial that you determine the nature of these needs precisely so that they can be satisfied as efficiently as possible.

As you may have guessed, the issues that must be confronted during such a study are not unlike those a systems analyst must wrestle with when designing an order entry system. For example, this individual would be concerned with the kind of system access data entry personnel require and the content of management reports. Quite clearly, different individuals often will require highly specific information which is of little consequence to others. On the other hand, recognizing common needs is just as important to the design of an integrated system.

In designing operations management systems, much the same is true. To truly centralize your information, operators will need access to the consoles of all systems. In addition, particular operators may only be interested in a subset of all console messages. For example, an operator charged with servicing tape drives will have a keen interest in tape mount requests, but will not be so concerned about interactive users logging on and off. A security specialist, on the other hand, should be aware of these logons, and will also want to monitor remote users logging on through dial-up modems. A tape librarian will profit from reports regarding the frequency in which particular tapes are used, and from knowing when these tapes are needed during the production schedule. By monitoring console messages based on function, a shop would be better able to optimize its resources through specialization and centralized control.

To design a proper operations management system, you will need to identify the flow of information within your company. You should ask yourself what types of messages pass through your console on a day-to-day basis? What kind of information flows from your users to your operations staff and technical support? Such analysis will help you to identify the information which is critical to your operations, and the person who should receive these messages. It is only when you have a good understanding of your environment that you should move towards the type of centralized operation referred to earlier.

WHY CENTRALIZE?

For the same reasons that you want to divide console messages into logical categories for your operators, you should aim to structure your DP department in a manner that reflects the logical structure of your installation. In multi-machine environments, machines are rarely isolated from one another with respect to your company's production. Most shops network their machines in order to ease the transfer of information. There are usually many dependencies among machines for production, so the independent management of each HP3000 is an inefficient way to control your DP environment. It would be better to have centralized control and management of multiple systems so that all information can be easily related. The result is that DP managers can make better management decisions and increase overall network uptime because they are working in a more controlled environment.

Some of you probably have, or are already taking steps towards centralization today. Batch scheduling software is a good example of this. Third-party vendors such as OCS and Unison offer utilities which allow you to do sophisticated batch job scheduling and which give you the ability to have cross-machine job dependencies. By knowing the status of jobs on all of your systems at one central site, you can truly run a distributed network efficiently. When provided with up-to-date information about the status of all your systems and by having all production scheduling information routed back to a central site, you can schedule jobs throughout a network and load balance across a number of systems. This means a maximization of the use of your resources by having all systems available from one central location.

Centralization also has implications on system security. By having control over all consoles, you are reducing the number of access paths to your systems; that is, fewer people need the high level capabilities required for operating your machines. Secondly, since all console messages are centralized, it is easier to watch for unauthorized access (i.e. logon violations can be grouped together for easier identification.)

Centralization of your consoles will also allow you to centralize your MIS staff -- both operators and specialists. If all machines can be accessed and full console control is available from one central location, then you don't need to have duplication of staff at remote sites. Because commands can be issued centrally, your resources and expertise can be at one site.

TOWARDS AUTOMATION

Once centralization has been achieved, you can begin to consider automation. It is important to note, however, that automation is a relative term. It should be viewed as an effort to reduce human intervention without necessarily eliminating it. So, the real issue is not whether to automate, but how much to automate. Significant gains in reduced intervention can be attained without substantial costs associated with automation and, in particular, artificial intelligence applications. It is also important to remember that automation comes with time; until you know exactly how and what to automate, you should not attempt to do it.

Before implementing an automation strategy, you need to know exactly what to automate. The key ingredient for such an analysis is information. A good way to gather this information is to log all your operations activity -- that is, the console traffic over a period of time. Next, analyze it to see what can be automated by considering the types of responses required for the messages coming in. Once this has been determined, you can design a plan and only then begin to implement procedures. Undoubtedly, you would have gone through a similar process in implementing a job scheduling package.

A job scheduling package is a good example of an automated process that shops "grow into" over a period of time. In many cases, system managers start out by using the MPE :STREAM JOBxxx;AT=time command to run jobs at night unattended. Running jobs overnight is often the solution to completing processing without affecting performance for on-line users. However, where job dependencies exist and jobs have to be run sequentially, using the STREAM command may not be enough. With STREAM you would have to guess when job A will complete so that you can stream job B to run once A has finished. If your predictions are off, jobs could be colliding or, alternatively, you could have a lot of idle time on your system.

At this point, a shop considers a job scheduling package to automate all overnight procedures. Once the scheduling package is installed, it is up to you to carefully plan your production so that processing continues properly. Typically, when people learn that the package is up and running, they will find more and more jobs that can be scheduled into it. Anything that can be automated will be automated over time. Most shops find that automating job scheduling makes life easier for operators as there is less of a need for human intervention. The result is an increase in overall productivity of machines and staff since available processing time has been optimized, and your operations staff's time has been freed for other tasks.

In automating your entire operation, the process for implementation is much the same: review what you are currently doing, analyze what can be automated, plan for implementation, and then implement. The process does not stop there, however. Just as jobs are continually added to a scheduling package, operations tasks should be continually evaluated as candidates for automation. After the initial implementation, your focus becomes one of review and analysis of tasks to be automated.

WHERE SHOULD ANALYSIS START?

Listing from your console will give you an accurate picture of what happens on a system from an operational point of view. From a log file, you need to identify which mechanical tasks are possible to automate. In many cases, either HP or a third party offers a way of automating a task to varying degrees (as in the example of batch job scheduling). Currently, there is a wide variety of software packages that can be purchased to help you automate: automatic error detection, high-speed backup, database manipulation, spooling, performance analysis, and so on. An in-depth study of console activity and your overall operational goals will help you to decide which products may be of benefit to your shop. The key to these tools when moving towards automation is that they are programmable, where expert systems and artificial intelligence may be closest to complete automation.

Of all the messages which are sent to the system console, those that require human intervention are perhaps the most critical. These could range anywhere from a reply to a tape request to correcting an error in a job to restarting a failed system. In each case, someone has to be made aware of the situation; otherwise, processing will not continue as planned. Rather than automating these tasks, you could put an alert mechanism in place. Depending on how critical each message is, the appropriate reporting mechanism could require a console command to be issued, a message to be sent to a specialist, or a programmer on call to be contacted. Through an alert mechanism, you will know how much human interaction is necessary and where. This is invaluable information for your operations automation plan since it can help you to determine your requirements for staffing and software tools.

Simply stated, the rationale for analyzing your console activity is to know exactly what to automate and how. This helps you make sure that all bases are covered when moving to automation. The more information you have about your console activity, the more situations you can account for. Again, going back to the job scheduling example, if you know what the various types of possible errors are, then you can set up a number of contingency plans to compensate for those errors. Another example of this is the automation of procedures relating to system failures.
Through analysis of your console activity, you will know the most common reasons for system downtime, and you can then set up procedures to restart your system, tailoring

the restart to the type of failure.

Also remember that automation is something that should evolve as you learn more about your system and your operation. In the early stages, the mechanism you use to alert personnel to critical situations may simply be a beeping message at the console. Later, it may grow to include a voice messaging system that automatically dials a programmer on call. To handle the problem of loading tapes and replying to requests, some companies are going so far as to get robots to mount tapes and then to program a reply which will execute automatically.

Even for less critical activity, it is a good idea to log information so that you can monitor the demand for certain types of tasks. Again, this will also help your specialists complete their jobs more efficiently. Consider the example of system performance. Both Hewlett-Packard and Carolian Systems are taking steps towards making it easier for users to centralize and automate performance analysis through the creation of an interface between a performance tool and the console. HP has chosen to tie OPT/3000 to their recently announced OpenView product, thereby allowing performance-related information to come back to a central site. Carolian has added a module to SYSVIEW so that the program can run in a "monitor" mode. In this mode, information about unusual performance situations gets passed back to a central console where a performance specialist can then monitor and respond to these messages.

IN SUMMARY

In order to manage multi-HP3000 sites effectively, centralization and automation of console activity are necessary. In order to develop an effective operations optimization strategy, however, careful analysis and logging of console operations is required. It is only once your objectives have been defined that you can begin to implement your plan. Such long term planning will be well worth the effort since centralization of all your HP3000s will mean increased control over your entire operation, optimization of both your hardware and human resources, and greater opportunity for standardization across your organization.

Once centralization has been put in place, a plan for automation should begin. Automation should be implemented over time so that you can automate as many different tasks as possible. Remember that any tool you choose to help you automate should be flexible, programmable and able to grow with you. Once again, the benefits of a well-planned automation program will become obvious as it frees up time for your operators, allowing them to concentrate on the tasks critical to your operation. As utilities become more and more sophisticated, you may find it possible to program even these tasks.

Reading through any HP publication, you will find that there is a growing need for standardization and efficient use of both hardware and human resources. It is this need which is driving users and vendors alike to search for and implement solutions which can incorporate both the control of centralization and the power of automation in their operations optimization projects.

$LOCALITY

Memory Requirements Planning for HP3000 MPE/XL Systems

by

Michael Hornsby

Beechglen Development Inc.
2026 Beechglen
Cincinnati, Ohio 45233
(513) 922-0509

I. Introduction

Why do the MPE/XL HP3000's need so much memory, and what are the issues for making the maximum advantage of all of it?

A. Initial Impressions

When the HP3000/930 was announced, my first thoughts were along the lines of what would MPE/XL possibly do with all of that memory. I thought of exotic random number generators to fill up all that space. Then shortly thereafter, the 935 and 950 obsoleted the 930 and it was announced that the standard memory required for a 950 would be 64M bytes, up from a mere 48M bytes. I thought that MPE/XL must have some really far out scheme for filling up all of this memory space, or that HP was attempting to corner the market on memory chips.

After much experience in converting systems from MPE/V to MPE/XL and tuning MPE/XL systems for maximum performance, the following is a discussion of the questions at hand.

B. Memory Constraints

The best way to think of memory's role in processing data is as a buffer on which the CPU can perform tasks. The buffer is broken down into two kinds of entities: data and instructions. The CPU reads the instructions and performs the requested action. No action can be performed on any data that is not in memory.

Generally, the faster the CPU is at performing instructions, the larger the buffer required to keep up with it, and the more concern there is with keeping the CPU busy. One essential inescapable fact governs all computing devices, and that is they all wait for something to do at the same speed.

Not having a large enough buffer to keep up with the CPU has two primary effects on the overall performance of the computer in question. First, a shortage of memory places an upper limitation on the computer's overall throughput, and second, some scheme is required to manage the staging of data and instructions into and out of memory from or to a peripheral storage device.

1. Limitations on Concurrency and Throughput

When analyzing system performance, what is actually done is breaking the system down into its individual component servers looking for the particular governing constraint. Thus, an HP3000 really can be thought of as:

CPU Servers and Co-processors

Disc Servers

Terminal Servers

Channel Servers

The role of memory is to buffer the interaction of these various work centers and allow them to work in consert to produce the desired result. The amount of memory available for holding code and data places an absolute upper limit on how many transfers can be done between servers in a given period of time.

The skill in performance tuning is to identify current or potential bottleknecks and perform appropriate tasks to reroute or alleviate them.

2. Paging and Virtual Memory

On an MPE/XL system, memory is divided into pages of 4096 bytes each. This allows the operating system to manage the memory contents by copying pages to and from disc as needed by the CPU in a process known as swapping.

As on MPE/V systems, code pages are never swapped out, only read in from their original program file on disc. Data pages are managed via the transaction manager and are swapped in and out as required.

C. Memory Overhead

The CPU and disc IO's required to manage memory are referred to as overhead since this activity does not contribute directly to the processing of data.

1. Swapping

Processes on all HP3000's are dispatched according to a priority scheme in which the lowest numbered priority is the next process to be launched.

When a process is launched, the first thing that happens is that all code and data pages that this process requires must be brought into memory, if not already there. If not, the memory manager swings into action to swap in from disc the required pages.

The idea of paging is to keep the most active pages in memory and remove the idle pages. This is done by a memory management process that scans through memory marking pages as overlay candidates. If on a second scan through memory a page is still marked as an overlay candidate then the memory manager knows that this page can be swapped out. Pages that are referenced in mid-memory management scans have their overlay candidate flag turned off.

When the memory manager finds free space or makes a space, then the process can start to do useful work. If however, no free space can be found or freed up, then the memory manager gives up and the next process with the highest priority is launched.

2. Dispatching

The task of finding the next highest priority process and initially launching it is delegated to a dispatching process. When a process is running and something it requires is not in memory, an I/O request is generated and control is returned to the dispatcher. Thus when a system has insufficient memory CPU, overhead is accumulated for the memory manager and the dispatcher, as both will be more active attempting to cope with demands for memory that must be accommodated.

The existence of a memory shortage can be directly inferred by the amount of CPU time spent on memory management and dispatching. Also, the benefits of additional memory or decreasing memory demand can be directly measured by these percentages.

3. Disc IO

Memory is used to buffer data from the discs because in relative terms it takes a long time to transfer data from disc to memory. On MPE/V systems, disc caching was added to the file system in an attempt to feed the data to the CPU faster.

This solution anticipates that the next blocks of a file are likely to be accessed and instead of bringing the particular block requested into memory, brings several blocks along for the ride. This works very well for programs that read files from front to back, but is not very useful for programs that read files randomly.

On MPE/V systems the CPU used to scan and manage the cached data was reported separately and ranged from 15% to 20% of the CPU. This was a good trade off if the number of physical disc I/O'S could be reduced by 60% to 70%.

On MPE/XL systems, disc caching is done at the hardware level. In other words, all files on the system are treated as virtual memory and thus the overhead to managing the pre-fetching of data into memory is charged to memory management.

So the main reason that MPE/XL systems require so much memory is that a bigger buffer is required to keep the faster CPU busy. A secondary reason is that due to the reduced instruction set the code generated for programs is ten to fifteen times larger than on MPE/V systems.

II. Differences between MPE/V and MPE/XL Systems

On the outside, running programs are very similar between the two architectures. However, behind the scenes there are major differences that should be understood and dealt with as part of the migration and conversion process.

On MPE/V systems, many expensive work arounds were required for implementation of large applications. These design strategies required under MPE/V may no longer be required under the MPE/XL architecture and may also cause significant performance problems as the number of users on the system are increased.

Unfortunately in most conversion attempts, the priority was on getting the applications converted and running with an absolute minimum of code changes. Thus the last thing on anyone's mind was application performance tuning, and in fact the prevailing attitude seems to be to wait until there is a performance crisis to invest any amount of time in performance tuning.

This is why MPE/XL performance issues are so dimly understood and so little performance tuning data is available. It's very sad to see people scrambling to cram 16Mb memory cards into their systems in the hope of a simple solution that only dents their pocketbook slightly.

A. MPE/V Run Time Environment

In order to contrast and compare the types of performance tuning tasks between MPE/V systems and MPE/XL systems,we need to compare the similarities and differences of the run time environments.

On MPE/V a process requires at least one code and data segment in order to exist. In actuality an average process would have many code and data segments.

1. Programs

On MPE/V a program is made of up variable length segments whose individual size could not exceed 16K. For efficient program execution you would segment programs at compile time with a $CONTROL SEGMENT= statement or in COBOL break the program up into sections. Initially, when HP3000's had small amounts of memory these segments were suggested to be about 8K bytes each. This allowed for easy overlay of one code segment with another.

The good thing about this scheme was that a branch could go directly to any location within the segment. The bad side of this scheme was that the entire segment, regardless of its size, would have to be swapped into memory, no matter how much of the segment was actually reinforced. This is one reason that the memory requirements of MPE/V systems increased dramatically with the number of different programs being executed.

2. Data

User data is initially loaded into a special kind of data segment called a stack. Other data segments are used for file buffers and additional storage. The maximum size of a data segment is 64K bytes.

For large applications the stack limitation was quickly reached and other extra data segments may have been used to hold overflow data. These large data segments have to be read in their entirety, so the memory requirements of MPE/V systems also increased dramatically if the applications being run required large stacks and extra data segments.

3. Limitations and Work Arounds

For large applications, the work around to the segment size limitations of 32K bytes for code and 64K bytes for data, was to create a new separate process and thus get a completely new set of code and data segments.

This option is very expensive in terms of system overhead for memory management and dispatching, but there simply was no other choice. For most large shops, process handling was the key to building large user friendly applications.

MPE/V disc caching was an adequate work around to improve I/O speeds. The improvements for batch processing were dramatic, and it extended the useful life of many HP3000's. Disc caching was done all in software and could be enabled or disabled for each disc drive individually. This allowed tuning by placing files that were heavily accessed serially on drives with disc caching enabled, and randomly accessed files could be placed on drives without disc caching enabled.

Overall, it was rather amazing that "classic" MPE lasted as long as it did from systems built initially for a small number of users with 512K of memory to a system the size of a Series 70 with 8M bytes of memory.

B. MPE/XL Run Time Environment

On the surface, running programs under MPE/XL is very similar to the MPE/V environment. However, it is very important to understand the fundamental differences in the linking, loading, and addressing methods used.

1. Programs

Programs executed under MPE/XL can exist in one, or a mix of three executable states:

Compatibility Mode

Object Code Translated Mode

Native Mode

Compatibility mode programs are created via the MPE/V compilers, and with a few exceptions can be restored from an MPE/V system and run on the MPE/XL system. The exceptions I have experienced are with data communications programs in the area of system to terminal, system to other HP1000 and 3000 systems, and system to IBM host.

When running in Compatibility mode, the operating system must look up each MPE/V instruction in a table to translate to the equivalent MPE/XL machine instructions. This process adds a great deal of overhead to executing the program.

Even if you have recompiled all of your applications into native mode your system may be spending a great precentage of time in compatibility mode if you are using KSAM, message files, Image logging, Sort/3000 and many other HP and non HP utilities, which have not been recompiled into native mode. See Figure #1.

After it was realized that the overhead for looking up and translating instructions would be unacceptably high, a utility was created that would pre-lookup the MPE/XL equivalent instructions and append the translated program to the end of the MPE/V compatibility mode program. This eliminates the overhead at run time but expands the program in sectors consumed from ten to fifteen times.

Native mode programs are created by the new MPE/XL compilers, COBOLII/XL, FORTRAN 77/XL, PASCAL/XL, and C/XL. If your source is in some other previously supported language then you'll have to convert it to one of the above to get the benefits of native mode.

A process created by MPE/XL has a minimum of two spaces: one for instructions and the other for data. A space is a virtual address that is referenced by a 32 bit space address and a 32 bit space offset. The code and data spaces are broken up into quadrants which are addressable up to one gigabyte each. The quadrants are selected and loaded into a particular register based on the two high address bits.

When a process is created, the loader will determine if the code spaces required exist and if not, create them. A separate code space is required for each executable library that is externally referenced. The data spaces for each code module are loaded as required in the data space.

2. Program Design Changes

Obviously the above discussion raises some questions regarding the best conversion strategies for existing programs and strategies for designing new applications.

The predominant theme of most conversion plans has been to get the applications running on MPE/XL with a minimum amount of effort, with very little regard for performance and system capacity requirements. The overall attitude is the MPE/XL system is multiple times faster and has multiple times as much memory, so why worry about capacity planning.

This is why many MPE/XL implementations have come up short of their expectations for performance, and most have come up very short of disc space.

III. Memory Management on MPE/XL Systems

The role of the memory manager on MPE/XL systems is to keep heavily accessed pages in memory and flush unreferenced data pages out to disc. Code pages are overlayed and reread when they are re-referenced.

The key to maximizing overall throughput and performance of an MPE/XL system is in understanding how to measure and interpret CPU overhead for memory management and how to improve program and data locality.

Locality is the art of grouping commonly referenced code and data together so that a small number of memory pages are referenced frequently.

A. Measuring Memory Management Overhead

When measuring performance on MPE/XL systems it is critical to keep in mind that the memory manager will seek to page code and data into memory and keep it there, until other more frequently referenced pages force them to be flushed or overlayed.

This causes very abrupt running of the memory manager, as new programs are executed and new files are opened. Thus to capture the overall effects of memory constraints and CPU overhead, we need to monitor the system over a long period of time. This is because we might catch the system in a loading cycle and generalize this peak to be an average.

The SURVEYOR program does a good job of reporting the short period (30 seconds to a few minutes) performance of the system, but it is best to monitor memory management overhead over much longer periods, typically 8 to 24 hours. This allows overall trends and cycles to be identified and analyzed.

Figure #2 shows overall system utilization for an HP3000 as the dispatcher sees it, Memory Management, Overhead (interrupts and other linear queue activity), Sessions (CS queue activities), and Batch (DS and ES queues). This system has 96 Mbytes of memory which is adequate because memory management CPU never exceeds 8%. However the amount of CPU spent in Overhead is fairly high at times. After some investigation it was discovered that a job management utility was unnecessarily running in the BS queue.

B. Improving Program Locality

In the rush to convert and recompile programs into native mode, very little attention is paid to optimizing programs for performance. This is especially true in two areas: process handling and program locality.

Applications that had to be broken up into multiple processes due to MPE/V segment size limitations can now be reassembled into a single application program.

These larger programs can also be tuned by grouping sections of code that are frequently accessed into locality sets using compiler directives.

1. Compiler Directive $LOCALITY

All MPE/XL language compilers allow grouping of code by using:

$LOCALITY name, for FORTRAN 77/XL LOCALITY option for PASCAL/XL LOCALITY pragma for C/XL PROGRAM NAME for COBOLII/XL

These directives enter the locality set groupings into the relocatible object module, and when the program is LINKED the code locality sets are grouped together.

2. Benefits of Locality Sets

The MPE/XL branch instruction has a range of 256K within the current code space. If the link editor encounters a branch greater than 256K, additional code is generated similar to an external call. This can degrade performance if these procedures are being called frequently. If a locality set is created that includes frequently called procedures, they will more likely be kept within the 256K limit.

Locality sets can minimize program page thrashing. On all MPE/XL systems, memory is broken up into pages of 4096 bytes each. When memory pressure is created by an intense batch job for example, unreferenced pages will be flushed out or overlayed. This can cause page thrashing for program pages that are occasionally referenced. This is one of the main reasons that MPE/XL systems have sluggish interactive response times while batch jobs are active.

Grouping code in a locality set tells MPE/XL to treat the locality set as a single page so that usage is based on the entire set, not just a single page. This can significantly improve interactive response times and reduce the overhead for memory management.

3. Locality Set Assignment Strategies

Whatever strategy you choose, keep in mind that locality sets should be kept as small and tight as possible, and that the best time to implement them is when the programs are initially converted over to native mode. There are two basic strategies that can be used depending on your knowledge of the application program.

The best method is to group procedures based on usage. Those procedures most heavily used should be grouped together. This requires some knowledge of the application, but I have found that it is possible to easily separate out those procedures that are infrequently used and then group the remaining procedures together.

Another method is to build a locality set of the core procedures and let the rest default. This method depends largely on the extent to which the program has been structured.

In either strategy, the benefits of supplying the memory manager with additional data can significantly reduce the amount of overhead required to manage memory.

C. Improving Data Locality

Optimized data locality is the basic hypothesis behind any memory mapping or disc caching algorithm. If the references to data are localized, then the overhead to read in, store and manage large chunks of files in memory will increase throughput by reducing the number of actual disc I/O's required.

If on the other hand the references to individual data records are very random and the file being processed is very large, then memory mapping and disc caching add little to the throughput of the system and are very expensive in terms of overhead.

The key to managing and improving data locality is to recognize that the system is attempting to fit the square peg of disc space into the round hole of memory. The degree of difficulty in performing this trick depends on the relative sizes of the objects and the tools at your disposal.

Figure #3 illustrates the relationship between the rate of page faults and the amount of Memory Management CPU required to bring the requested data into memory.

1. Disc Space and Memory Size Requirements:

The problem with the current set of memory planning formulas is that they do not take into account the number and states of applications and disc drives on the system. The formula that I have evolved is slightly more complex but yields a closer match to real life situations.

The formula is comprised of two parts, a factor for the user application state and population, and a factor the type disc I/O's and the amount of disc space on the system.

User-Application Factor (UA):

.90 - Straight CM Migration
.85 - OCT Major Applications
.80 - Mix of CM and NM Applications
.75 - NM Major Applications
.70 - CM Single Application System
.65 - OCT Single Application System
.60 - NM Single Application System

Disc-Space Factor (DS):

4 - Mostly Batch Applications
5 - Mix of Batch and Interactive Applications
6 - Mostly Interactive Applications

Memory Requirements Formula:

#MB = (#Users * UA Factor) + (#Discs * DS Factor)

For example a 100 user system with 6 disc drives running an interactive applications in a mix of Native and Compatibilty Modes would be:

(100 * .8) + (6 * 6) = 106 Mbytes of Memory

That same system running in Compatibility Mode would be:

(100 * .9) + (6 * 6) = 116 Mbytes of Memory

So the memory requirements planning process is really a two dimensional process, with a factor for user growth and a factor for data base growth.

2. Identification of Most Active Files

In order to maximize data locality in memory we must maximize data locality on the disc itself. The first step in this process is to identify all of the busiest files on the system. The second step is to make sure that these files are optimally stored on disc. And the third step is to reorganize the data within the files so that commonly accessed entries are physically adjacent to each other.

There are three basic ways to identify file usage:

By Inference of Size

From Partial Backups

Logfile Analysis

Busy files can be isolated for further attention by identifying the largest files on the system. The inference is that the data has to get into these files somehow. This can be done with a report command to first look for the largest groups, then the largest files within those groups.

A partial store listing will show only the files accessed on a day-to-day basis. The usefulness of this list varies from system to system.

Finally MPE/XL and MPE/V systems both allow for logging of file close records. These records can be analyzed to definitively pick out the busiest files on the system.

3. Using MPEX to Tune Files and Data Bases

Once the list of files that account for 90% of the Disc I/O's are isolated, they can be optimized using MPEX or any other tools you may have for reporting and reblocking.

The first step is to make sure that these files are blocked efficiently or else the wasted space at the end of each block will also waste memory when paged in.

The second step is to identify the image data sets in the list of busy files by doing a LISTF, DB. This shows whether it is a master or a detail and how full it is.

Master data sets should be no less than 35% full. Master data sets are usually accessed randomly and mostly empty masters mean that many pages of binary zeros are being read in.

Empty masters also slow down serial reads because the bitmap on the front of each block needs to be read in and scanned to see if any entries are in that block.

Detail data sets on the most frequently accessed list may benefit enormously from packing related entries on a chain together. This can be determined by further analysis with DBLOADNG or HOWMESSY. If these report that the average chain length is greater than three and is spread across multiple blocks, then the time to pack this path will probably be justified.

4. Using ADAGER to Create Data Base Locality

Once the specific data base tasks have been isolated you could unload and load the entire data base to and from tape. This is a very time consuming process. I have had great success with ADAGER in performing these tuning tasks. The only unfortunate part is that most shops have tools like MPEX and ADAGER and can benefit greatly from their use, but lack the guidance and/or instruction on their proper and timely use.

By changing capacities of heavily accessed master data sets so that they are not too empty, we are creating greater locality of data in that master.

By packing the most frequently accessed detail data sets along the path most frequently used, the locality of the detail is increased by a factor of the average chain length. This is probably the most significant task that can be done to improve performance on any model of HP3000. The lasting effects depend on the volatility of the data entries in the data set.

Figures #4 and #5 are histograms of the performance improvements made by simply packing detail data sets over two successive weekends.

5. Image Buffer Specifications for MPE/XL Systems

The buffer specifications for any of the data bases that have a member data set on the most frequently accessed list should be increased to the maximum.

The default buffer specifications for data bases are from the archaic days of systems with 128K to 512K of memory. On any MPE/XL system there is enough memory to allow for the complete opening up of the buffers for the most active data bases.

The rough maximum number of buffers can be calculated by dividing 27,000 by the size of the largest block. So if the

BLOCKMAX was 512, the number of buffers to specify would be 54. This is done in DBUTIL by entering the following command:

>SET dbname BUFFSPECS = 54 (1/120)

This tells Image to build a global area with 54 buffers for 1 to 120 DBopens.

IV. Conclusions

A. Guns versus butter decision

So all of this discussion on locality and performance tuning comes down to a decision to either commit to weekly, monthly and quarterly maintenance or to attempt to add 16M byte memory cards until you max out the box or run out of money.

In my experience it is always best to spend the time to study your applications and develop a maintenance plan that will maximize the overall performance of your particular system.

B. Implications for migration planning

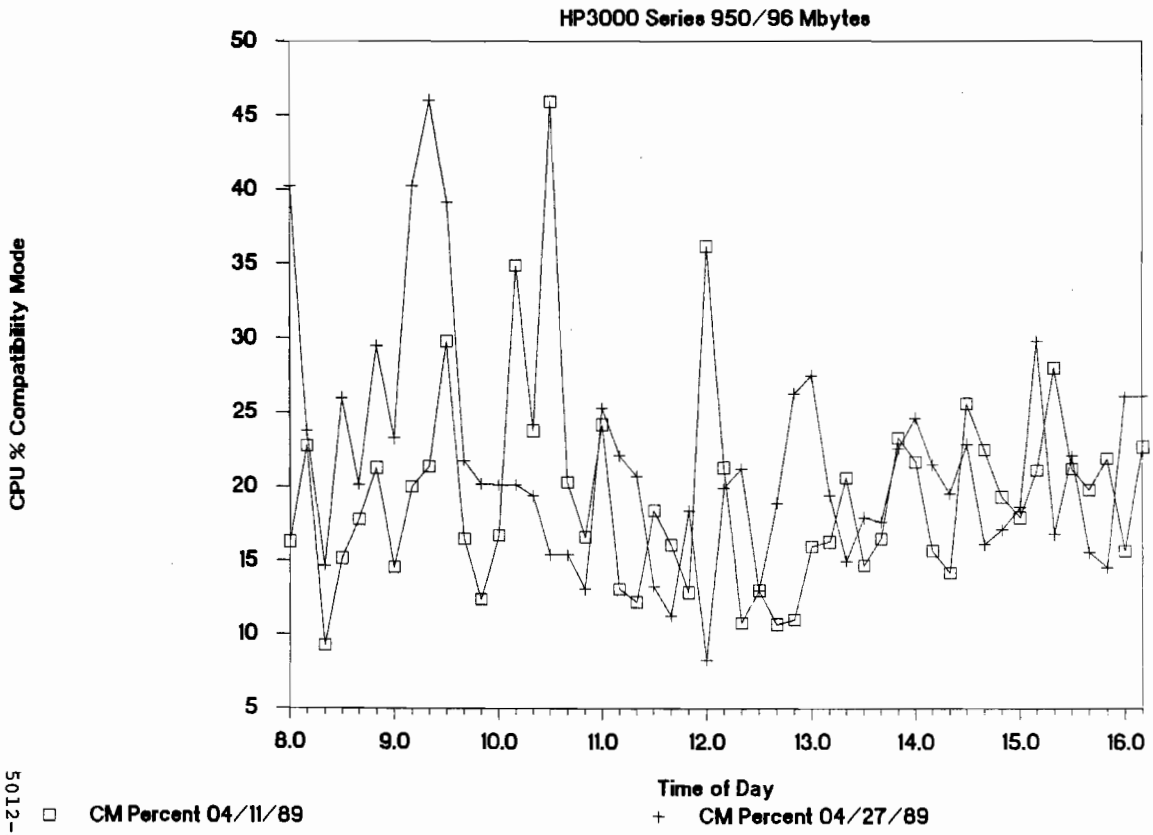
If you are planning on converting to an MPE/XL system, do not rely on batch benchmarks for performance. These can be very misleading for all of the reasons documented above. Don't let yourself get caught in the squeeze between promised performance improvements and budgeted spending constraints.

When planning the conversion process, allow time for unimplementing some of the constraining MPE/V "solutions". It will be much less expensive to remove these time bombs at conversion time than to wait until you desperately require every ounce of performance the system can deliver.

C. Identifying and scheduling maintenance procedures

Maintenance procedures, just like system backups, need to be scheduled and performed on a regular basis to be of value. The types of procedures and approaches to these procedures take on a large importance on bigger systems. However, because there is more horsepower available, these procedures can be performed on a more frequent basis.

Figure #1



5012-15

Figure #2

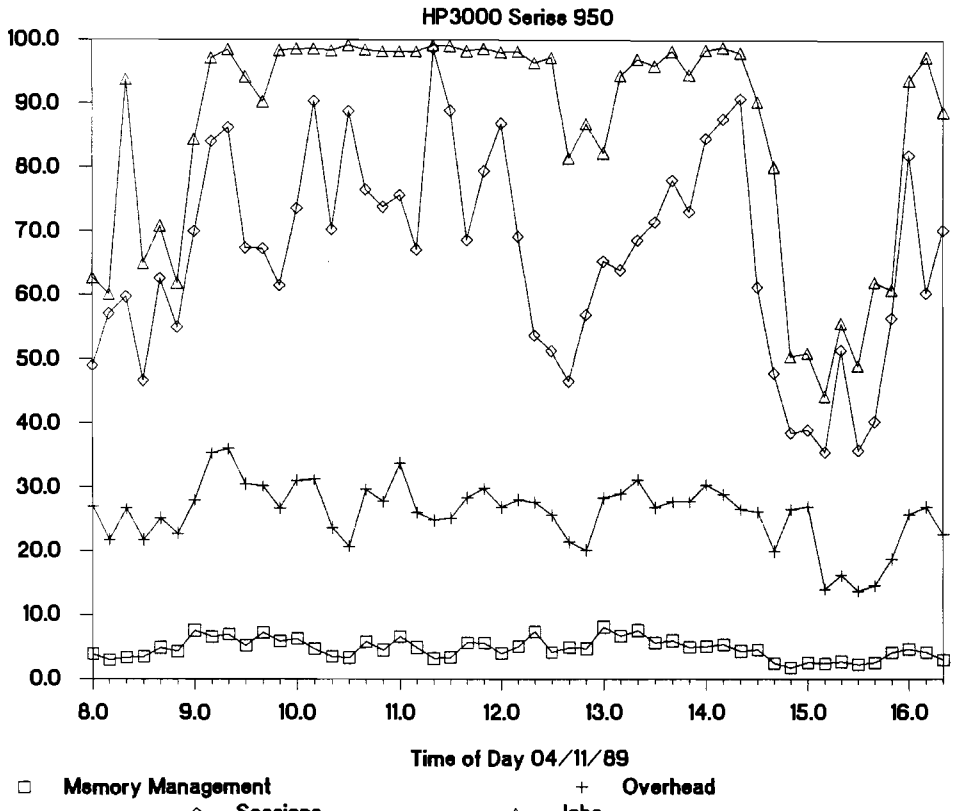

HP3000 Series 950
CPU Utilization
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
8.0
9.0
10.0
11.0
12.0
13.0
14.0
15.0
16.0
Time of Day 04/11/89
Memory Management
Overhead
Sessions
Jobs

Figure #3

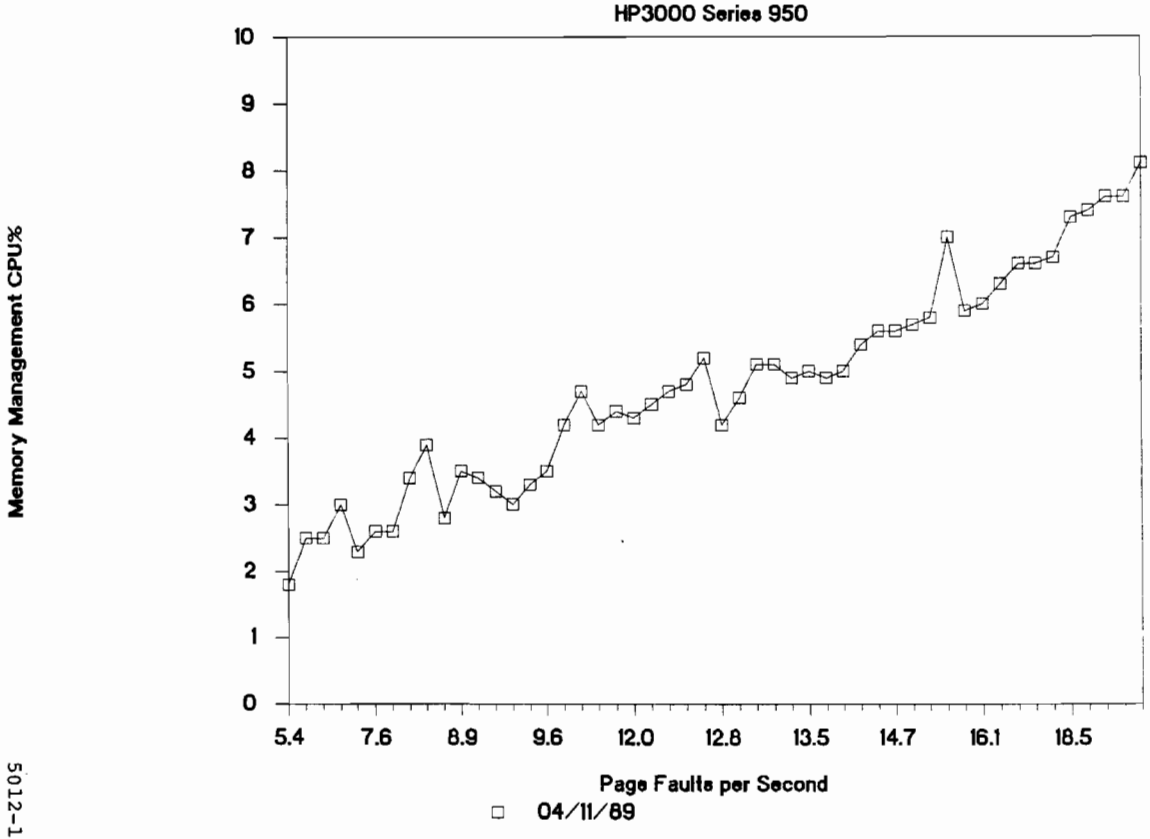
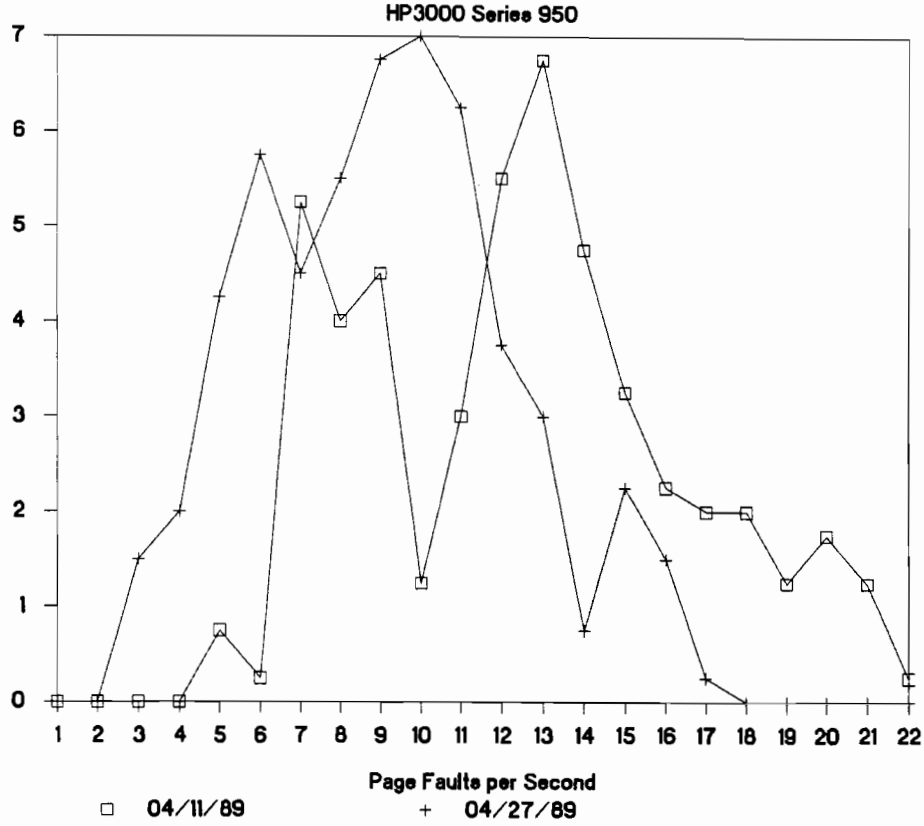
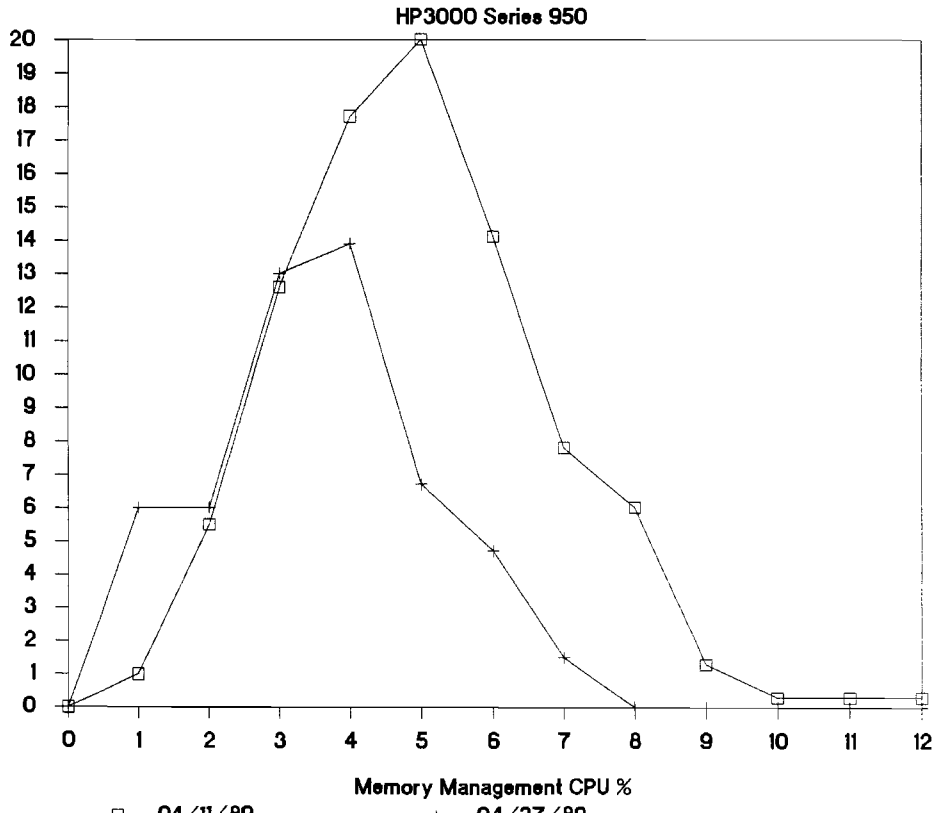

HP3000 Series 950
Memory Management CPU%
10
9
8
7
6
5
4
3
2
1
0
5.4
7.6
8.9
9.6
12.0
12.8
13.5
14.7
16.1
18.5
Page Faults per Second
□ 04/11/89

Figure #4



5012-18

Figure #5


HP3000 Series 950
Event Count
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
0 1 2 3 4 5 6 7 8 9 10 11 12
Memory Management CPU %
□ 04/11/89
+ 04/27/89

TUTORIAL:
DATA PROCESSING OPERATIONS
by M. E. Kabay

JINBU CORPORATION
P. O. Box 509 Westmount, QC
H3Z 2T8 CANADA
(514) 931-6187 FAX: (514) 931-0678

OPERATIONS: MANAGING DIVERSE RESOURCES
People
Software
Hardware
Disaster Prevention

PEOPLE:
The Operations Team
Communications with the User Community
Documentation for Less Work

SOFTWARE:
Data Integrity
Security
Accounting Structure
Performance Monitoring
Database Administration
Operating System Tuning

HARDWARE:
Equipment
Facilities

THE OPERATIONS TEAM
Customer Service
Structure and Function
Technical Training
Client Support
The Redundancy Rule
Users as Part of the Team

CUSTOMER SERVICE
Psychology of Your Users
--Who are your users?
--Why do they choose you?
--What determines continued use?
--What makes them happy?

STRUCTURE AND FUNCTION
Use TIME-SLICES to determine responsibilities
Assign JOBS to FUNCTIONS.
Evaluate AGREEMENT on roles.

TECHNICAL TRAINING
HP courses
Other external training
In-house training
Training forever

CLIENT SUPPORT
Operations staff may be first-line client contacts
Psychology may be more important than technical issues
Train your staff to handle crises

THE REDUNDANCY RULE
No information shall be known to only one member of the team.

USERS AS PART OF THE TEAM
Think of US not THEM.

COMMUNICATIONS WITH THE USER COMMUNITY

Outreach--users as system monitors
A friendly, helpful CATALOG.PUB.SYS
Follow-up: a key to cooperation
Sharing knowledge: a prime motivator
:JOURNAL, your electronic bulletin board
The WELCOME message
The team meeting

OUTREACH--USERS AS SYSTEM MONITORS

Organizing Meetings With Users
Discuss with senior management
Hold seminars in users' area
Informal style
Be sure DP staff mingle

AGENDA FOR MEETINGS WITH USERS

What operations do for users
Whom to contact for help
Why immediate comment is useful
How system management can correct problems quickly for users

A FRIENDLY, HELPFUL CATALOG.PUB.SYS

Encourage favourable feelings about the system
Convey necessary information pleasantly
Explain important errors and what actions to take
Make problem reporting easy--PSCREEN and phone
Give operators critical information fast

FOLLOW-UP: A KEY TO COOPERATION

When a User Reports a Problem
take responsibility
define the problem
write down the details
report back to the user

SHARING KNOWLEDGE: A PRIME MOTIVATOR

What Encourages People to Learn?
Immediate application
pleasure, not pain, active involvement

NON-DIRECTIVE TEACHING

Avoid classical lectures (like this one...)
Avoid answering questions: help participant fiture it out
Benefits: self-respect, questions encouraged, more enjoyable for all

:JOURNAL, YOUR ELECTRONIC BULLETIN BOARD

```
\ENTRY=MENU
*** WELCOME TO JINBU'S BULLETINS
***

:FILE INPUT = source
:FILE CATALOG = JOURCAT;SAVE
:RUN MAKEGAT.PUB.SYS,HELP

JOURNAL
OPTION NOLIST,NOBREAK
FILE
CICAT.PUB.SYS=JOURCAT.DOC.SYS
HELP
RESET CICAT.PUB.SYS
***
```

THE WELCOME MESSAGE

Useful info
Bulletin update
Important news
How to indicate problems fast
Security warning to permit legal action against vandals

THE TEAM MEETING

Weekly: supervisors
Monthly: everone possible
Topics: review events and plans
Benefits: sharing knowledge, coordination, team spirit

PEOPLE
Documentation for Less Work
Will this have to be done again?
What to document in JCL and how
UDC catalogs
:HELP catalogs

WILL THIS HAVE TO BE DONE AGAIN?
If so, document what you do:
part of systematic problem solving
useful as part of training
Enter reference in problem database:
save time by not re-inventing solutions
ensures others have access to info

DOCUMENTATION
Database for problems and solutions
KEY_____ PRI__
ON ___ WHO _____
BY ___ WHAT_____
OK ___ MORE_____
_____
_____(NOTES)_____

WHAT TO DOCUMENT IN JCL
name of JOB
dependencies and resources
effects on other jobs
mileposts within job
never use UDCs in job

HOW TO DOCUMENT JCL
:COMMENT
:EDITOR
:FCOPY or LIST
:TELLOP & GETOPREP

DOCUMENT YOUR UDC CATALOGS
Use meaningful parameters
OPTION NOLIST,HELP
:COMMENT
LIST program in NOLIST UDCs
:TELLOP
FCOPY, APPEND to log info

SOFTWARE
Data integrity
Security
Accounting structure
Performance Monitoring
Database Administration
Operating system tuning

DATA INTEGRITY
Backups
Magnetic tapes
System log files
Memory log analysis
Preventing disc failures
Known-good coldload tapes

BACKUPS
A Unique Copy Of A File
Is Not A Backup
Costs of backups
Application vs system
Check your backups
Fast backup software
Unattended backup
Backup before HP PM
Double backup before RELOAD

MAGNETIC TAPES
tape library management
acquisition and turnover
cleanliness
storage

SYSTEM LOG FILES
can log everything if necessary
costs I/O (small) and filespace
audit trail, error log, console
valuable for optimization
use TELESUP utilities (LOGAUDIT, LOGREPT)
useful in chargeback systems

MEMORY LOG ANALYSIS
HP predictive maintenance checks
use MEMLOGAN to verify
report bad chips to HP

DISC-RELATED FAILURES
CSMON
from HP: alerts console of bugs
should run all the time
HP Predictive Maintenance
will identify problems
follow their guidelines

KNOWN-GOOD COLDLOAD TAPES
Make a new CLT only to change something
Update from KGCLT before making new tape

SECURITY
Fundamentals
MPE security
Security auditing features
Security risks in MPE
HP solutions
Third-party solutions

WHAT'S THE PROBLEM?
People vs user IDs

MPE SECURITY
Logon
Capabilities
Resource limitations
Access restrictions
Hierarchical overrides
Run-time verification
IMAGE/3000 security
Backup and logging

SECURITY AUDITING FEATURES
Console messages
System log files
Database log files

SECURITY RISKS IN MPE
Capabilities (SM, OP, PM)
Logon assistance
Password limitations
Readable directory
Batch job files
Unattended terminals
Programmatic access to MPE
Logon UDC failure
Invisible failed FOPEN
Diagnostic tools

HP SECURITY SOLUTIONS
HPSECURITY
Logon control
Password control
Hierarchically restricted access to MPE
Audit trail

THIRD-PARTY SECURITY SOLUTIONS
VESOFT'S SECURITY/3000
OCS/PRIVATE & OCS/KEY
INCASE'S ENGARDE
TELAMON'S CONSOLE ENGINE
other products

ACCOUNTING STRUCTURE
Security
Resource accounting
SYS account
Canonical passwords
System UDC catalogs

ACCOUNTING STRUCTURE: SECURITY
User ID
Session ID
Local attribute

RESOURCE ACCOUNTING
Few accounts or many?
logon/logoff
:REPORT info
file access
directory limitations

SYS ACCOUNT: PARTITION
PUB
DOCUMENT
UTIL
UTILPM
CONTROL
COLDLOAD

CANONICAL PASSWORDS:
Change Default Passwords
Vendor accounts
TELESUP and other HP accounts
Use jobs for temporary access

SYSTEM UDC CATALOGS
Few and simple

PERFORMANCE MONITORING
Users' feedback
Definitions
Verification by timings
Monitoring tools
Performance by design

DATABASE ADMINISTRATION
Capacity
Pointers
Tools
Design

TUNING YOUR OPERATING SYSTEM
Resources:
tables, queues, caching, spooling
Timing:
daily, weekly, monthly, yearly

HARDWARE
Equipment
Facilities

EQUIPMENT
Records
Configuration: meaningful classes
Preventative maintenance
Lease/Buy
New/Used
3rd party solutions

FACILITIES
Data Centre Appearance Affects Credibility
Clean, neat, businesslike, safe
Secure, efficient, planned, controlled

DISASTER PREVENTION
People are the key

ELECTRONIC FORMS AND THE HP3000: ANOTHER STEP ON THE ROAD TO THE AUTOMATED OFFICE

By Richard J. Armitage
Business Systems International, Inc.
20942 Osborne Street
Canoga Park, California 91304
(818) 998-7227

Forms are a good tool. They allow information to be categorized for ease of interpretation. They standardize and simplify the communication of complex corporate and legal data. A properly designed form can highlight information that would otherwise be lost in a sea of numbers.

Form provide value, and that is why we use so many in business and government. Yet, like any good tool, forms -- at least the traditional preprinted forms -- cost money:

- money to design
- money to print
- money to store

These are the obvious costs associated with using preprinted forms, but what about the other less obvious costs -- the ones that tie an organization to inefficiency and waste. Let's briefly review some of these costs.

First, there is the personnel cost for time spent finding the right form and then mounting and aligning it for printing. There is also the potential of using the wrong form. Here the cost is wasted time to re-run a job, or it no one detects the error, embarrassment or even a lawsuit.

Next, there are costs associated with managing preprinted forms. These include time and energy spent controlling access to forms, monitoring inventory levels and reordering forms, and making sure that only the current version of a form is being used.

The cost of obsolescence is a major consideration in using preprinted forms. As requirements change, forms must change; and the old forms aren't worth the paper they are printed on. (Murphy's Law suggests that requirements for a form will change just after you order a large supply of the old form.)

Perhaps the most serious problem associated with using preprinted forms is the cost of keeping inefficient or outdated forms. These are the forms which should be redesigned, but are not because of the cost of creating, printing and implementing a new version.

Today, with the technology of laser printing and the sophistication of available software, there is no reason for a company to incur the excessive cost of buying preprinted forms, or to continue to be tied to this outdated technology. Electronic forms are here, and they represent a significant step on the road to the automated office.

In this presentation, I will discuss the electronic form as an alternative to the preprinted form. I will cover the capabilities of the laser printer which make electronic forms a viable alternative to preprinted forms. I will then outline considerations involved in developing electronic forms, merging data with electronic forms in an HP3000 application environment, and finally I will review some actual cases involving the implementation of electronic forms. Throughout the presentation I will attempt to point out the advantages in cost savings and improved productivity resulting from the switch to this new technology.

First, what is an electronic form? Viewed very basically, an electronic form is nothing more than a file containing instructions which allow a laser printer to print an image of the form. The instructions are stored in a host computer and sent to the laser printer's memory for merging and printing with application data.

Let's briefly review some of the advantages which electronic forms offer the modern business office. We can begin with the obvious costs is time and money spent on designing and printing forms. With a preprinted form it can take weeks to design the form, do the artwork, review proofs, make changes, and start the process all over again. Once the form is finalized, there is the actual cost of printing, shipping, and storing the preprinted form. With an electronic form you can create and begin using a new form in a matter of hours. You can fine tune the design without incurring additional costs other than your own time. There is no need to order and pay for a year's supply, you print the form only when it is needed. Because the electronic form is stored in the computer, it is always ready for use in each company location. You have eliminated the problem of wasted time spent looking for a form or inadvertently using the wrong form. And when the form is printed, it is done with laser printer quality and perfect alignment of data every time.

We spoke about the effort spent in managing preprinted forms. With electronic forms, this effort is reduced significantly. Since forms are stored in the computer, you no longer have to worry about physical access control. You can restrict access to your electronic form files just as you restrict access to your computer programs. Certain forms can be restricted to individuals or departments, while others are universally available. You do not have to be concerned with inventory levels or lead times for reordering forms. Most important, there is only one version of the electronic form available -- the current version. You can forget about tracking down supplies of obsolete forms spread throughout the company.

Since forms are only printed as needed, there is no large inventory on hand when you decide to change a form. The redesign and implementation of a new electronic form is easy and straightforward. The day the new form becomes effective, it is the only form available for downloading to the laser printers.

One more benefit before we move on. With an electronic form it is easy to print the output on demand, and in the users' department. You do not have to batch the output until you have enough to justify the time to mount a new form, nor do you have to print all your forms centrally because your users cannot mount and align forms on their own department printers.

Electronic forms technology is a significant advancement over traditional preprinted forms technology. As we look more closely at the technology, you will note that it not only offers cost savings, but it also offers tremendous flexibility in the way you design forms based applications. I will discuss several of these design considerations throughout the presentation, but first let's review some of the basic capabilities of the laser printer. I will focus on the Hewlett-Packard LaserJet Plus family of printers.

LASERJET CAPABILITIES

Early in 1988, Hewlett-Packard shipped its one millionth LaserJet printer. While most of these printers are being used with personal computers to produce high quality word processed documents, more and more data processing professional are recognizing the LaserJet's special capabilities, and they are installing these printed as spooled output devices for their HP3000 users.

Some of the special capabilities of the LaserJet Plus family of printers which make them especially suitable for electronic forms applications are:

1. Print Quality

The LaserJet prints at 300 x 300 dots per inch resolution which comes very close in quality to typeset documents. It handles both graphic images and text on a single page, and allows a variety of fonts to be used on a page.

2. Fonts

The LaserJet printer offers an almost unlimited variety of fonts, with both fixed and proportional spacing. The available point sizes range from as small as 4 point up to 72 point.

Fonts are available as either resident fonts which are permanently stored in the ROM of the printer, cartridge fonts which plug into cartridge slots in the printer, or soft fonts which are stored on the host computer and can be downloaded into the printer's RAM memory.

3. Programming

The LaserJet provides a control language which can be used to communicate from the host computer application to the printer in order to access the printer features. Hewlett-Packard's printer command language (PCL Level 4) has become a laser printing standard. PCL commands are also referred to as escape sequences. These commands are used to control cursor movement, font selection, line drawing, definition and placement of graphic images, etc.

4. Memory and Storage

The LaserJet printer has its own memory which, in addition to buffering input and building a page image, is used to hold fonts and macro files which contain PCL commands which direct the printer. The advantage is that once fonts and form macros are downloaded and stored in the printer's memory, they are available for use throughout the day. Thus, in most situations, only data has to be transmitted to the printer during a production job.

The LaserJet offers these and many other capabilities which makes it ideal for handling a variety of print jobs in the user department. You can put a LaserJet Series II into a user department for less than $2,000. The printer is equivalent to a 300 line per minute printer, yet it is quiet, easy to use, and extremely durable. With the LaserJet IID you get the added benefit of additional paper trays and duplex printing.

CREATING AN ELECTRONIC FORM

If you analyze most business forms you will notice that they typically contain lines, boxes, shaded areas, text, and graphic images. Most of these components can be defined with PCL commands. With enough experience, anyone can design even a complex form using PCL commands, fonts, and graphic images. For example, to draw a horizontal line the escape sequence has to specify the starting cursor position, the line thickness, the length, and the dot pattern. The PCL command to draw a line would look like this:

```
{ESC}*p300x400Y{ESC}*c90a150bP
```

It is not my intention to teach you how to write PCL escape sequences. There is an abundance of software which will create these PCL commands for you. It is, however, important in selecting and using such software that you to understand which components of a form can be drawn with simple PCL commands and which need to be graphic images, and the difference between storing and using a form in raster format or vector format.

First, let's review the components of a typical form and identify those specified with PCL and those which must be graphic or scanned images.

* Horizontal and Vertical Lines - PCL
* Boxes - PCL using horizontal and vertical lines
* Shaded Areas - PCL
* Text - fonts
* Special Characters - some with fonts, others must be scanned images.
* Logos - scanned images
* Diagonal Lines - graphic images
* Reverse Text - scanned images or special fonts

There is a significant difference between using PCL commands and graphic images. PCL commands can be stored and processed very efficiently. Graphic images, on the other hand, take up a lot of space and are slow in processing. Since some very popular electronic forms software as well as many desktop publishing packages make extensive use of raster graphics, you should understand the significance of defining and storing a form in raster versus vector format.

I will begin with an explanation of raster. Raster refers to images which are composed of groups of dots. When a laser printer prints an image it creates the image by laying down a pattern of dots (90,000 for each square inch).

These dots are actually specified by individual bits of data, each bit being either on or off. These patterns of dots can be communicated from the host computer to the printer in either their dot format (raster) or as vector commands which the printer converts to dots at the time the image is printed.

In raster each individual dot (or bit) is transmitted from the host computer to the printer. It requires a large amount of information to describe even a small raster image. For example, a one square inch box would require 90,000 bits (300 x 300) or 11,250 bytes of data. A complete 8.5 by 11 inch form would require over one million bytes of data.

In vector an image is communicated to the printer in a command format similar to the line draw command previously described. The printer converts the commands into a dot pattern just prior to printing.

While raster provides ultimate versatility and flexibility in defining laser output, its use should be limited to those situations where vector commands are not available. Using PCL vector commands to define an electronic form saves transmission time from the host computer to the printer, and requires less printer memory to store the form. It also saves disk space on the host computer.

If you plan to purchase forms design software, there are a number of other features of which you should be aware. First, remember that a form is designed only once, but filled in many times. If a form is to be filled in with computer generated data, you must be concerned with how easily the package allows you to register the form for data, and how easily the data merging part of the package allows you to integrate the electronic form into your computer application. Ideally, you should be able to implement an electronic form with little or no modifications to you application program. (I will review data merging requirements in the next section.) Some other factors to consider are:

* Equipment required for forms design. Many packages require a 386 bases PC with a VGA monitor.

* Ease of visualizing and printing during design.

* Control over fonts and the number of fonts included with the package.

* Alternatives for downloading fonts and forms.

* Alternatives for activating forms.

* Support for duplex printing.

* Support for multiple part forms.

REPLACING PREPRINTED FORMS WITH ELECTRONIC FORMS

As I mentioned before, a form is designed once, but it if filled out many times. Understanding the data merging requirements for electronic forms is important if the electronic form is to be successfully implemented. Data merging is the process of merging data from a host computer application with the electronic form.

In many ways, using an electronic form with an HP3000 based application is essentially the same as using an impact printer and preprinted forms. First, the form must be available when the data arrives at the printer, and second, the positioning of the data on the form is typically handled by the computer application.

Actually, the elements needed for successful merging of data from an HP3000 application with an electronic form are:

1) The form to be filled in must be in the printer's memory.

2) Any fonts required by the form or the data must be available.

3) The form must be activated for overlay with the data.

4) The data from the host computer must be available.

When and how the printer is prepared for data merging depends on your requirements. Again the key steps that must be handled are: send the soft fonts to the printer, send the form to the printer, and activate the form.

One approach is to have your application handle all three steps. This can be accomplished in one of two ways. First, use a custom environment file which contains all of the soft fonts and the form required by the application, and have it included in the spool file with your data. The other method is to send the fonts and forms to the printer as separate job steps, and then activate the form for overlay either by a terminal type file with the appropriate escape sequence or by having your print program send the escape sequence prior to the data. Once a form is activated, it will be overlaid with each page of output data until another form is activated or until the printer is reset.

The advantage of this approach is reliability, that is, all required elements will be available when needed. The main disadvantage comes in if the application runs several times throughout the day or if the same fonts are used by several applications. In that situation, there is excessive overhead involved in sending fonts and form files to the printer each time the application runs.

Another approach for handling printer preparation in situations where forms and fonts are used on a routine basis is to store the soft fonts and the forms required by each of your applications in the printer's memory so that they are available whenever needed. The LaserJet Series II can store up to 32 macro files and the LaserJet IID and the LaserJet 2000 can store over 32,000 files. Under this approach you establish host computer job streams to download the fonts and forms required by each laser printer. The jobs are executed each time the printers are powered up. The fonts and forms stored in the printer's memory as permanent remain resident as long as the printer is powered up.

With the forms and fonts already stored in the printer, your application only has to send the escape sequence to activate the appropriate form for overlay. Again this can be accomplished by use of a terminal type file, or by modifying the application to send the escape sequence prior to the first output record of data.

To work properly, the forms and fonts should be stored in a centralized library on the HP3000 and unique numbers assigned for downloading. This avoids conflicts between users and duplicate storage of the same fonts with different numbers.

USING TERMINAL TYPE FILES

Hewlett-Packard's terminal type files are a valuable aid in using laser printers connected to the HP3000 through an asynchronous interface. These files allow you to send a sequence of PCL commands to the printer in the same spool file with your data, without having to modify the application program. This is important because in most spooled printer environments a reset is automatically sent to the printer at the beginning of each spool file. The reset returns the printer to its default environment and turns off the form macro overlay. Thus, when an application requires you to activate a form, the escape sequence to accomplish this must be sent to the printer in the same spool file as the data. With terminal type files you can accomplish this without modifying the application program. You merely use the ENV option of the FILE command to access these terminal type files.

HANDLING COMPLEX MULTIPLE PART FORMS

Multiple part forms are perhaps the chief challenge to electronic forms software. The laser printer allows you to print up to 99 copies of any output record, however, the copies are identical. In the traditional multi-part preprinted form set, each part may be slightly different or even completely different. Routing designation and areas of data excluded are examples of some typical differences. In addition, the back side of each copy could be different. While even the most complex multi-part form can be handled if you have the ability to do form switching from within your application, situations where a third party application cannot be modified are far more typical.

The challenge in the multiple part forms environment is to process a single output record to several copies of the same or different forms. Our approach is to create a customized environment file with the multiple part processing requirements. The multiple part form will actually reside in the printer memory as a single form overlay, however, each part may have a unique routing designation printed on it and be routed to a different department printer. In addition, duplex printing may be designated for some or all parts, and the contents of the back page may be different for each part. With this capability, the LaserJet will be able to produce forms which replace even the most complex preprinted form sets.

IMPLEMENTATION OF ELECTRONIC FORMS - CASE STUDIES

In this section, I will cover three situations where electronic forms have been implemented with great success. The first situation, which was described in an article in the November, 1987 issue of The HP Chronicle, involves the Central Coast Computing Authority (CCCA) in Santa Barbara, California. CCCA is a public agency with data processing responsibility for the Santa Barbara Community College District, as well as two school districts comprised of 24 high schools, junior high schools and elementary schools. They service approximately 130 users, most of whom are at remote locations.

CCCA's initial application of electronic forms was for printing student transcripts. Under the old method, when a student or parent requested a transcript, the member school would transmit the request on-line to CCCA, where it was held until off hours for printing. At that point it was necessary for the computer operator to initiate the job, mount the preprinted transcript forms, and then print the small number of requested transcripts (usually from 1 to 20 each night). After printing, the transcripts were sorted and sent to the requesting schools either via mail or courier. Considerable time delays occurred routinely, and a number of people were involved in the process.

The implementation of electronic forms for this application was simple and straightforward. Most of the schools already had laser printers on site, so the actual printing of the requested transcripts was shifted from the central facility to the individual requesting schools. Here is how the system works. The file containing the electronic transcript form is maintained in an master electronic forms library on the central HP3000 computer. Access to the form and the font files is via an on-line request screen which allows each school to request that these be downloaded to its laser printer. When a transcript is requested, the job is submitted on-line from the remote location. The job is no longer deferred for overnight processing, instead it is processed immediately, and the transcript data is sent to the laser printer at the school where it is merged and printed with the electronic form.

The application was easy to implement, but the benefits were significant. Now instead of days, the turnaround time from request to printing is just minutes. Also, most of the man hours have been eliminated from the process, since the computer operator and the mail room are no longer involved.

By eliminating the special handling required by the preprinted form, CCCA was able to have the transcripts printed at the user location. This is a typical example of electronic forms replacing preprinted forms, with overnight batch processing (which is very typical for applications requiring preprinted forms) being converted to a more responsive on-demand, real time approach.

Another example of electronic forms speeding up processing and eliminating man hours is a billing application at a large Southern California manufacturer. This company processes thousands of invoices each day on a high speed line printer using preprinted forms. In addition, each day they receive several requests for duplicate copies of invoices. The initial application of electronic forms involved printing these duplicate invoices.

Under the old approach, a copy of the data for each invoice was maintained on microfiche. When a duplicate copy of an invoice was required, a clerk would print the microfiche copy, and then retype the information on a blank invoice form. The company did not want to send xerox copies of invoices. Thus, the time spent just locating the microfiche copies of invoices and then retyping each on an invoice form added up to several hours each day.

The new system involved developing an electronic form and writing a custom program to access a file of invoices. The program accesses the required invoice data and merges it with the electronic form, which is almost identical to the original preprinted invoice. The program is run on-demand by an accounts receivable clerk, and the copy of the invoice is printed on a LaserJet Series II in the accounting department.

Here again, the implementation of the electronic form reduced the time spent on this activity, and resulted in faster turnaround time.

Another example of electronic forms is an application involving the TDY travel order form for the U.S. Navy. This application involves the use of both an electronic form and electronic signatures.

The Naval Ship Weapons Systems Engineering Station at Port Hueneme implemented an on-line financial management system which includes a number of very sophisticated on-line approval processes. One of these involves the generation of TDY travel orders, which allow government personnel to receive advances for business travel expenses. The system allows for the requesting, approving and authenticating of travel orders to be done entirely on-line. However, once the actual travel order is printed, each of the three officials involved is required to sign the form. At Port Hueneme about two hundred different supervisors and managers are authorized to sign travel orders, and over 50 travel order requests are processed and printed each day.

Prior to electronic forms, the electronically approved travel orders were printed in the travel office on preprinted forms, and then someone would carry them around the base getting the appropriate officials to sign the orders. The implementation of electronic forms and electronic signatures was designed to eliminate this duplicate effort, and to streamline the process of generating travel orders.

The system operates on a Hewlett-Packard Vectra computer which is located in the travel office and connected to a DEC computer which runs the financial system. Printing of the travel orders is done on a LaserJet Series II. Approved travel orders which are to be printed are transmitted to a signature merge program operating on the Vectra in the travel office. The Vectra was used for the signature merge program for two reason: (1) the financial system on the DEC could not be modified, and (2) the users wanted access to the electronic signatures and the signature merge software to be physically restricted.

The names of the requesting and approving officials on the travel order record are used to search for the appropriate electronic signature images, which are stored in encrypted form on the Vectra's hard disk. The travel order data, the signature images, and the electronic form are merged and printed by the laser printer.

The system saves time and money, but more importantly it allows the Navy to take advantage of the control and approval functions built into the financial system. The electronic signature image on the physical document is proof that the travel order was in fact electronically approved.

CONCLUSION

We have covered a lot of ground in our discussion of laser printers and electronic forms. Hopefully, we have succeeded in demonstrating that the laser printer has the potential to be an invaluable tool capable of assisting you in achieving tremendous cost savings and efficiencies within your office. At this point, I would like to recap some of the key advantages to be gained in using electronic forms in place of preprinted forms.

Reduced cost is one of the primary benefits to be gained with electronic forms. The direct cost associated with using preprinted forms are high and they continue to go up. The indirect costs of storage and wasted time are also significant. All of these costs can be eliminated or drastically reduced with electronic forms.

Electronic forms provide you with almost complete control, something that is impossible with preprinted forms. Access to any particular form can be restricted to certain people or departments, and the current version of an electronic form is the only one available for use throughout the entire organization.

Printing of the forms can take place in the user departments without worrying about inexperienced people having to mount and align preprinted forms.

Finally, you never run out of electronic forms.

The bottom line is that the laser printer has opened the door to a new office technology -- electronic forms. With this technology, you can make your business forms the efficient tools they were intended to be.

REPORT WRITERS - CURRENT TRENDS AND FUTURE DEVELOPMENTS

Roger W. Lawson
Proactive Systems Inc.
339 South San Antonio
Los Altos
CA 94022
(Tel: 415-941-9316)

ABSTRACT

Easy reporting of information held in central M.I.S. data bases is an important factor in the cost/benefit equation of most computer systems.

The options available to users are now expanding rapidly in a field that was previously fairly static so far has HP3000 computer users was concerned. With advanced menu systems, PC integration, natural language and other technologies rapidly moving in to take over from (or extend the capabilities) of traditional command language products on commercial systems.

The author will review the likely future developments from the perspective of a software vendor in this field and the impact they will have on the typical commercial installation.

The extension to integrated charting, and the use of graphic arts techniques on reports printed on laser printers will be looked at to see how this can fit in with reporting system strategy bearing in mind the likely growth in the low cost availability of such printers.

REPORT WRITERS - A PERENNIEL TOPIC

Report writers are of perenniel interest to commercial computer users. Why? - Because in many installations over 70% of the computer resources may be taken up in producing printed reports or terminal displays for informational purposes.

Every system managers life would be easy if he could get rid of report production or just optimize their processing. Every MIS Manager would save a lot of staff if he could really place report creation solely in the end users hands.

In reality the power and capabilities of report writers are increasing rapidly so there are very few cases now where a 3GL language is required to produce a report. However users needs have also become more diverse and more complex so that life for the software developer is not getting easier. A single report writer is unlikely to cover all the needs in future (I certainly don't know of any that supplies all the things I will cover later in this paper).

Report writers are therefore heading off in different directions - some emphasising ease of use, some emphasising facilities, some emphasising performance, etc. Let's see why, referring to the needs of HP3000 users.

USER NEEDS

Who are the users of report writers? The original report writers, e.g. RPG, QUERY on HP3000s, were designed for use by the software professional (or possibly because of their lack of user friendliness they were only capable of being used by such people). However if you examine real user needs they tend to fall into three categories:

- End Users. Typically they want to produce relatively simple reports quickly - often they are one-offs. Examples of such users may be middle or higher management, secretaries and clerks. The time to learn how to use the software must be short (management don't want to spend more than the absolute minimum of time and lower level staff turn over rapidly). Also the software must be capable of being learnt by unintelligent staff. Therefore complex facilities or features are not likely to be of much interest to this group. Similarly such users are often happy with default layouts of data columns and headings - they will not want to take the time to manipulate layout.

Recently a menu based approach has become popular to meet the needs of such users.

- Semi-technical Users. Typically these are staff who are willing to spend some time learning a report writer and are capable of doing so but they may not be trained computer experts. Examples would be accountants, engineers, senior secretaries, etc. Command driven report writers such as QUERY, QUIZ, etc may be acceptable to such users.

■ Software Professionals. These guys want lots of facilities so that they can always produce the information requested. They are often producing reports that will become part of some routine application processing which will be run daily or weekly - so efficiency of processing and the ability to optimize record retrieval at a low level is important. They may need to interface it to other software and the ability to control and modify output layout is important. Trying to meet the needs of all three of the above in one product is not sensible. It's like trying to produce an automobile that's friendly enough for grandma to drive around the block, that can beat a Ferrari in a quarter mile standing start and also has every conceivable gadget on it you can think of. It just ain't possible to do it and still deliver a product that is going to cost less than a million dollars.

SYSTEM MANAGERS NEEDS

One other person involved is the computer system manager. His concern is keeping computer resource usage within reasonable limits, balancing conflicting user needs, controlling routine production processing, etc. He has a big interest in the performance of report writers.

PERFORMANCE ASPECTS

Report writers fall into two categories - interpreted and compiled. Traditionally, on HP3000 computers, interpreted report writers have been more popular - this is probably because they are easier to produce by software developers and they were therefore out in the market first with relatively sophisticated features. However when compared with a compiled program they do not have as good a performance - see Figure 1 for some figures.

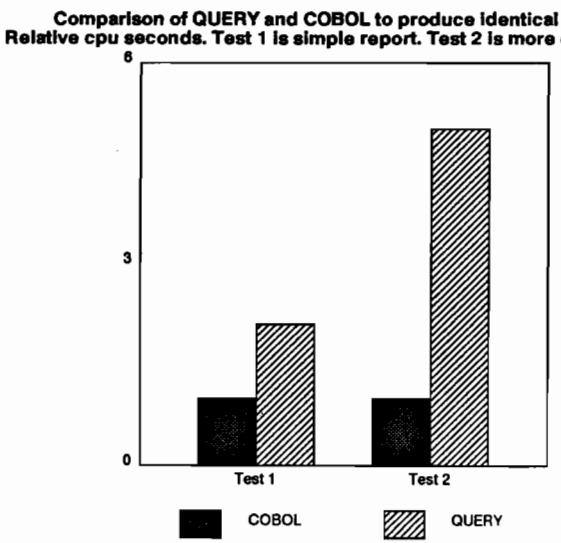


Figure 1

So if you want to improve performance certainly you should look at compiling your reports - compilers for several interpreted report writers are now available and this trend will certainly accelerate.

Secondly look at low level software aspects:

■ Does it run in native mode on MPE/XL systems (and does it generate native mode code if it's a compiler)?

■ Does it have the option of using MR,NOBUF reads on data bases thus bypassing the overhead of access via the data base software?

■ Does it optimize record retrieval automatically, or at least let you analyse and tune data base access? For example how does it handle cross data set joins in IMAGE?

One interesting feature that may be available soon with the introduction of multi-processor HP RISC machines is the possibility to have multiple processes running in parallel to produce a single report.

Another way to optimize some reports is to improve record indexing - reference the third party solutions that are becoming more popular to assist IMAGE record retrieval). These can help you with the "finding the needle in a haystack" problem even if they don't help much with a lot of routine accounting reports. If you don't want to spend money then it's often a good idea to look at adding extra paths (eg. add an automatic master in IMAGE or add an index in SQL). However faster reporting tends to have a trade-off in slower updating unless you split your data base into two replicated copies - one for on-line updating and the other for reporting. This is certainly now possible and you can even put the two copies on separate computers.

PC PROCESSING & DOWNLOADING

Another way to improve performance is to off-load work to those cheap PC computers. These have been used for firstly defining reports (ie. handling the report definition/compilation stage) and secondly for actually processing the information after extraction from the main system - HP ACCESS is a typical example of such an approach.

In addition with the growth in PC populations and the increased use of spreadsheet and graphics on PCs it has become essential to be able to produce at least Lotus 1-2-3 format files for download and processing on a PC (and there are many other possible PC formats also of course).

MORE FEATURES & MORE COMPLEXITY

Let's look at some of the other things that are becoming more popular.

■ Menus for everything. As exemplified by HPs BRW. Certainly I think it is now generally accepted that menus make life easier for the non-expert user. Whether they are the ultimate answer to all report writing needs remains to be seen. There comes a point where it takes longer to step through all the menus than it does to write commands - see natural language below also. Would turning the COBOL language into a set of menus really make it easier to use?

■ Dictionary driven. Dictionaries are really essential in my experience to make life easier for end users and software professionals alike. I have seen so many data bases where items are named AB123X, or similar. Certainly item alias names and alternate column headings are a must. To do a true multi-lingual data base and reporting system you also need multiple aliases (or multiple dictionaries). Even more messy are cases where multiple items are defined (eg. 10 weekly sales figures are defined as a single X100 item that contains 10 packed decimal numbers). Another example which is quite common but even more difficult to handle is where the same item name is used for different purposes in different data sets or tables!

■ Easy passing of data from one report to another. Look at Figure 2 and you will see that to calculate the percentages, a first pass to form the totals must be made before reprocessing the data. This can be done by supporting a disc output format that can be re-input, or by some programmatic mechanism to pass working storage values (the latter is a lot more efficient of course).

SALES BY PRODUCT

DESCRIPTION	VALUE	PERCENT
CHALK	157.50	11.91
SHOES	271.25	20.51
FOOTBALL	318.55	24.09
TENNIS RACKET	124.45	9.41
TENNIS BALL	107.35	8.12
BASEBALL BAT	342.80	25.93
Final Total	1321.90	100.00

Figure 2

■ Load control. If you go to many computer installations as I do, and ask them why they don't let users write their own reports, they often reply: well we tried that but we found they overloaded the system. Several report writers now enable system managers to control report writing load. For example the system manager can force users to submit reports in job mode and/or limit the size of serial data base reads.

■ Output pagination. Being able to produce a report and then simply browse through it on a terminal easily without printing it is a very useful capability. For example being able to select "first page", "last page", "next page" or "previous page". With 132 column terminals now common, this facility can reduce the need to print many reports. It can either be done by integration into the report writer or there are other stand-alone third party products that can pick up files in the spooler and display them.

■ Integration with electronic mail services. Being able to deliver a report to a user directly into his mailbox via a mail system can likewise save printing and postage costs.

■ Multiple data base and hardware support. The ability to support more than one data base and file type is becoming important, e.g. the ability to access HP IMAGE, HP SQL, and other data bases plus the ability to run on multiple hardware platforms is becoming important. As a result many report writers are moving towards a relational view of data bases, e.g. IMAGE sets are simply viewed as tables and the software works out how to join them (but look out for performance implications inherent in this approach).

With distributed systems becoming more common, the ability to retrieve from multiple data bases on different nodes is also required. Even efficient retrieval from remote data bases can be a problem at present using standard NS/RDBA facilities on NS. Again there are ways to speed this up using NetIPC and various other techniques. Query optimization in a distributed system where there are often replicated copies of data bases is an area for much future development.

The above are just some of the improvements that are becoming popular and which in most cases can be grafted onto existing report writer products. More revolutionary matters are covered below.

NATURAL LANGUAGE

Some people argue that the English Language is the ultimate user friendly interface. Why bother with command languages or menu systems when you can simply interrogate the data base directly? The user types in a request in English and receives the answer back in a tabular form (some systems can graph the output also). Commonly the interface to the data base is via SQL. An example of the translation from english to SQL that was performed by a prototype system we have developed is shown in Figure 3 (see the paper given at the San Francisco Interex conference for more information).

```
QUERY: Print a list of sales sequenced on stock code within
       supplier and showing supplier name and product name

SQL COMMANDS: select distinct products.descript,
                  sales.account, sales.stock_cde,
                  suppliers.supplier
                  from inventory, products, sales, suppliers
                  where products.stock_cde = sales.stock_cde
                  and sales-stock_cde = inventory.stock_cde
                  and inventory.supplier = suppliers.supplier
                  order by supplier asc, stock_cde asc
```

Figure 3

Notice how concise the english is in comparison to the SQL! There are some commercially available products that also claim to do this. To date, user take up has been limited partly due to performance problems and partly due to the set-up effort - you tend to have to spend several man weeks teaching the natural language system about the contents of your data base. However my firm belief is that within 5 years these products will be both technically and commercially viable.

For end users they could be a good solution but I am not sure that they are the ultimate solution for software professionals. But note that they often produce SQL which is becoming the lingua franca of data base access.

SQL STANDARDIZATION

With SQL rapidly becoming a de-facto industry standard (for example HP have now provided read access to IMAGE data bases from SQL), it is obvious that many products will use SQL as the underlying data access mechanism.

However don't imagine that SQL is much use as a report writer by itself. You need a lot of pre and post processing to match the report writing capabilities of current products.

GRAPHICS INTEGRATION

All the above will help in technically delivering the reports users currently ask for. But what about unsatisfied needs - graphics is certainly one of them. Many printed reports could be improved enormously by including line, bar or pie charts at suitable points. If this could be done easily, and without much cpu overhead which has been the problem with graphics in the past, then it would be a real step forward.

LASER PRINTING FOR GRAPHICS ETC

The advent of low cost laser printers has transformed the ability to produce graphics economically and quickly. For example even a standard LaserJet Series II can print integrated text, data and graphics at several pages per minute.

The other big advantage is you can produce really high quality output easily. For example look at the price list in Figure 4 - using proportional fonts and a two-column layout makes it look really good and you save paper as well - all you need is the software to make production of such reports easy.

However it does introduce more complexity - no longer is there a fixed pitch of 10 characters per inch and a fixed line spacing of 6 lines per inch. With nice proportional typefaces the horizontal spacing is totally dynamic (it even depends on the words/data on the line) and vertical spacing can also be totally variable. But don't panic - professional graphic designers and printers have been coping with this for a long time - it just requires different thought processes and a different technical approach.

Now there are products on the market already that can post-process output from existing report writers to do this but wouldn't the obvious answer be to integrate them together. Similarly chart production and the "electronic forms" support could be integrated. There are already products in the PC market that do this.

WIDGET SOFTWARE PRODUCTS INC. - PRICE GUIDE

Effective 1 March 1989 *Page 1*

Product No.	Description	Price ($)
Language Compilers		
1002346	Pascal Compiler for IBM PC Compatibles	340.00
1002352	Pascal Forms Management Utility	125.00
1010008	Cobol Compiler	555.00
1010009	Cobol Forms Management	75.50
1020044	Basic Interpreter	390.00
1020045	Basic Compiler	450.00
1020046	Basic Compiler Run Time System	99.00
1020047	Basic Compiler Forms Management	240.00
1030031	Fortran Compiler	350.00
1030031	Fortran Graphic Extensions	99.00
1030032	Fortran Language Extensions	75.00
Data Base Management Systems		
1102345	Superbase Relational DMBS	670.00
1102348	Superbase Run Time System	230.00
1102349	Superbase Management Utility	145.50
1102350	Superbase Forms Interface Compiler	340.00
1102351	Superbase Report Writer	265.00
1102351	Superbase SQL Language Module	288.00
1102352	Superbase Run Time Optimiser	350.00
1102353	Superbase Extended System Option 1	199.00
1102354	Superbase Extended System Option 2	199.00
1102355	Superbase extended System Option 3	199.00
1102356	Superbase Reconfiguration System	75.60
1102357	Fastbase Network DBMS	475.00
1102356	Fastbase Distributed System Option 1	240.00
1102359	Fastbase Distributed System Option 2	240.00
1102345	Superbase/2 Relational DMBS	670.00
1102348	Superbase/2 Run Time System	230.00
1102349	Superbase/2 Management Utility	145.50
1102350	Superbase/2 Forms Interface Compiler	340.00
1102351	Superbase/2 Report Writer	265.00
1102351	Superbase/2 SQL Language Module	288.00
1102352	Superbase/2 Run Time Optimiser	350.00
1102353	Superbase/2 Extended System Option 1	199.00
1102354	Superbase/2 Extended System Option 2	199.00
1102355	Superbase/2 extended System Option 3	199.00
1102356	Superbase/2 Reconfiguration System	75.60
1102357	Fastbase/XL Network DBMS	475.00
1102356	Fastbase/XL Distributed System Option 1	240.00
1102359	Fastbase/XL Distributed System Option 2	240.00

Prices are subject to change without notice. Prices include delivery to US mainland locations only. Contact your Widget Software Products representative for a specific quotation.

Figure 4

POST PRODUCTION EDITING

Similarly users often want to manipulate or edit report output. Integration with word processors/editors or spreadsheets would be another possibility (in the latter case an approach already taken by more than one company was to design their report writer as a spread sheet production facility).

SUMMARY

Well, after you've been through all the above you will realise that report writers are no longer the simple beasts they used to be. Life is getting more complex isn't it! The likely result is that computer installations will run more than one product and the market for report writers will generally become more fragmented. Certainly it is likely to result in the users having more freedom and choice in terms of the facilities available to them even if they are constrained by the usual limits of processor capacity. One thing I am sure of though is that the software professionals won't be out of a job.

REFERENCES

Hill S. (1988). A Report on Report Writers. Interex Orlando Conference Proceedings.

Lawson R.W. (1987). Comparative Performance of Report Writers. Interex Las Vegas Conference Proceedings.

Lawson R.W. (1989). Natural Language: The Ultimate User Friendly Interface. Interex San Francisco Conference Proceedings.

DESIGNING USER INQUIRY SCREENS FOR KEYED ACCESS

PAUL EDWARDS
BRADMARK COMPUTER SYSTEMS, INC.
1506 ESTATES WAY
CARROLLTON TX 75006
(214) 242-6660

DESIGNING USER INQUIRY SCREENS FOR KEYED ACCESS

ABSTRACT

The design of user interface inquiry screens is complex. There are many factors to be considered during the design phase. With the use of IMAGE or KSAM, the design is focused around a single key value to be used for the inquiry.

With the increased use of keyed access systems, the time devoted to design of the user interface is increased. These systems provide partial key, generic key, keyword, and relational access. This causes an additional work load on the systems analyst to define the real access needs of a user that does not understand these access systems or his own requirements.

An in-depth discussion of these access methods will be provided and practical examples of inquiry screen styles will be presented.

DESIGNING USER INQUIRY SCREENS FOR KEYWORD ACCESS

The design of user inquiry screens has been done much the same way since the HP/3000 environment of IMAGE and VPLUS has existed. The traditional access methods will be reviewed, access key types will be defined, the different types of access available will be discussed, application design requirements will be outlined, and some screen design examples will be described.

TRADITIONAL ACCESS. IMAGE and KSAM are two MPE subsystems used by most applications software developers to provide random and serial access to data. There are several disadvantages in both systems.

IMAGE allows only one key in each master dataset and that key has to be unique. There is no generic or indexed sequential key retrieval. To provide additional keys in a record, a system of automatic/manual masters and details must be implemented. This adds additional structures to a database that can degrade performance and complicate the program access of the database. Sorted chains are used to retrieve data records sorted by other fields. The updating of these chains can cause serious degradation in performance of the application program. IMAGE doesn't provide any relational access capability.

KSAM provides generic key and indexed sequential retrieval of data records. There are performance considerations in the use of KSAM with a large number of users. It doesn't provide the logging and security features of IMAGE. Key file corruption is possible with power/system failures. Key files must be checked for integrity after these events occur. Many people combine IMAGE and KSAM file structures to overcome the generic access limitations of IMAGE. This causes a more complex program environment with the possibility of loss of data integrity and causes more disk I/O.

Besides the performance considerations, the serial access of IMAGE and KSAM files have several disadvantages. The serial access of a master dataset can consume considerable time because all records are read whether they contain data or not. A detail dataset has a delete chain that is used to add records in the empty spaces left by deleted records. If the delete chain is empty, the records are added to the end of the dataset. The result is that a serial read of the dataset will retrieve records in no particular order. Even if a chained read is used, the records are returned in order by their placement on the chain. A sorted chain could be used to ensure a certain sequence, but with the associated performance overhead. KSAM records are stored in chronological order. Access of these records can be by any of the keys which is sorted in order by the key or in chronological order.

KEY DEFINITIONS. The types of keys available are simple, concatenated, grouped, and keyword. Custom and independent indexing can be done, as required. These types of keys provide different ways to access the data records. A B-Tree structure is used to store the keys and pointers.

The simple key is usually a single value in a field. This is the type of field normally used in KSAM or IMAGE. An example of this type of key would a customer number.

Several simple keys can be combined together to form a concatenated key. The order in which they are combined together are significant when retrieval is done by a keyed access system. Alphanumeric sorting order is left to right. An example of this type of key is an invoice number combined with the invoice line item number.

Multiple fields that are all to be searched at retrieval time are called grouped fields. They can be simple or keyword fields. Two address lines are an example of grouped fields.

A keyword key is created for each individual word in a field that is separated by spaces, slashes, dashes or any special characters. To save disk space, a keyword exclude file is available to enter common words that don't need to be keyworded.

Custom keys are used when additional intelligence is required to create the key. A special procedure is used to prepare the key. Data type conversion, reformatting dates, upshifting, embedded keys, and stripping unneeded characters are examples of key preparation.

Independent indexing is used for other than IMAGE databases. This includes MPE flat or word-processing files. The key datasets only contain the key values and any additional values required to be passed to the calling program.

TYPE OF ACCESS. The types of access to records in a keyed system are serial, partial key, generic key, keyword, relational, and boolean.

Serial access is in sorted sequential order. Access of the indexes by traversing the B-Tree will return the entries in ascending or descending order.

Partial key retrieval used when only part of a value is significant. A wildcard character is used to fill the places that are to be ignored in the search. An example would be a search value HEW??TT that would retrieve any entry starting with HEW followed by any two characters followed by TT.

Generic search is used to find entries beginning with

certain characters. A wildcard character is used to signify that all following characters are not significant. BRAD@ will find all entries starting with BRAD with zero or more characters following.

Since keys are created for each word in a keyworded field, the access is not positional. Keyword access is not case sensitive, either. These characteristics enable the user to locate values without regard to location or upper/lower case value. Generic or partial key access can be utilized, also.

Relational access is very powerful. It can provide access with multiple values across multiple fields, datasets, and databases using dynamically-joined indices. A common field in two datasets provide the link between records in each dataset. If a common field is not available, then a projection is made using a common field in a third dataset.

Boolean expressions can provide logical choices to be made qualifying entries. The AND, OR, and AND NOT operators are be used for making multiple criteria retrievals. A method of reduction retrieval can be used against a set of retrieved entries to reduce the number of entries displayed. This is achieved by making multiple DBFINDs against the same set of data records and changing the criteria each time.

APPLICATION DESIGN. Application design tasks are more complex with keyed systems due to the virtually unlimited ways to access the data. Two areas that need to have special consideration are intimate knowledge of the data and in-depth analysis of the retrieval requirements.

Knowledge of the data includes the normal field sizes and field data type descriptions with several other requirements concerning the contents of each field. The type of key, as defined above, must be determined for each keyed field. Any concatenations or grouping of fields must be specified. The length of the key is important so as not to use too much disk space and to provide enough length to make the key as unique as required for inquiry. How the data is represented in the field will determine patterns for partial lookups. For the keyword fields, a minimum number of characters per word will determine which words are keyed. The size of each word and the average number of words in a keyworded field will affect the B-Tree structure.

The retrieval requirements are more complex than with IMAGE or KSAM. The type of access required for each field or key must be selected as defined above. Since boolean relationships are available between different keys, these relationships must be defined. A thorough knowledge of the user's manual systems must be obtained as with traditional systems. In addition, the user's knowledge of his data retrieval requirements and education level is very important. Since any field in an IMAGE database is now

accessible, the user could very quickly be overcome by the complexity of the resulting screen design. The KISS (Keep It Simple, Stupid) principle should be followed.

SCREEN DESIGN. A sample database with sample screens for a simple customer invoice system is illustrated in the copy of the slides to follow. A traditional approach will be shown with the addition of a keyed system design added.

DESIGNING
USER INQUIRY SCREENS
FOR KEYED ACCESS

CONTENTS

- TRADITIONAL ACCESS
- KEY DEFINITIONS
- TYPE OF ACCESS
- APPLICATION DESIGN
- EXAMPLES

TRADITIONAL ACCESS

- IMAGE
  SINGLE/UNIQUE KEY MASTERS
  SORTED CHAINS
  AUTOMATIC MASTERS
- KSAM
- PERFORMANCE CONSIDERATIONS
- SERIAL ACCESS DISADVANTAGES

KEY DEFINITIONS

- SIMPLE
- CONCATENATED
- GROUPED
- KEYWORD
- CUSTOM
- INDEPENDENT

TYPE OF ACCESS

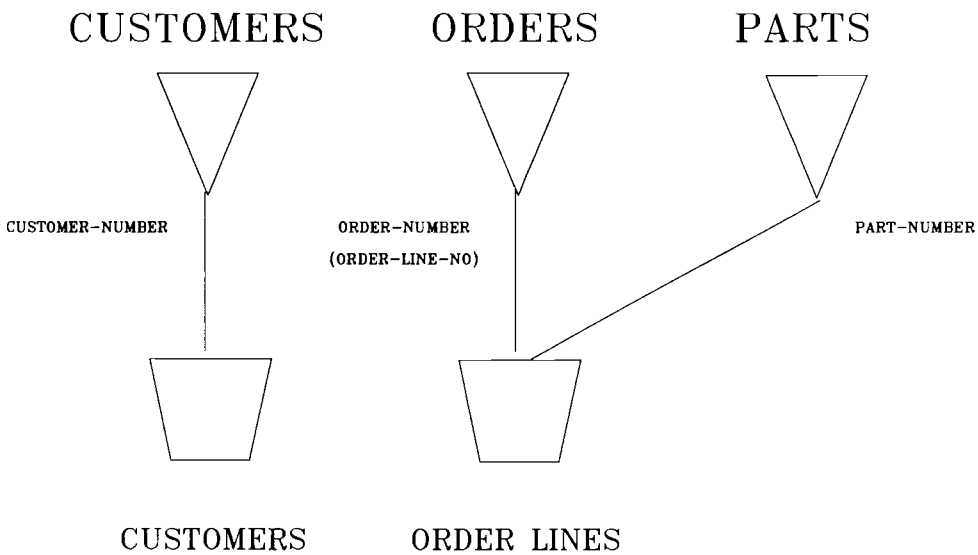
- SERIAL
- PARTIAL
- GENERIC
- KEYWORD
- RELATIONAL
- BOOLEAN

APPLICATION DESIGN

- KNOW YOUR DATA
  TYPE OF KEYS
  KEY LENGTHS
  FIELD CONTENTS
- ANALYZE RETRIEVAL REQUIREMENTS
  TYPE OF ACCESS
  BOOLEAN RELATIONSHIPS
  KNOW YOUR USER

CUSTOMER ORDER DATABASE

TRADITIONAL ACCESS



CUSTOMER ORDER SCREEN

TRADITIONAL ACCESS

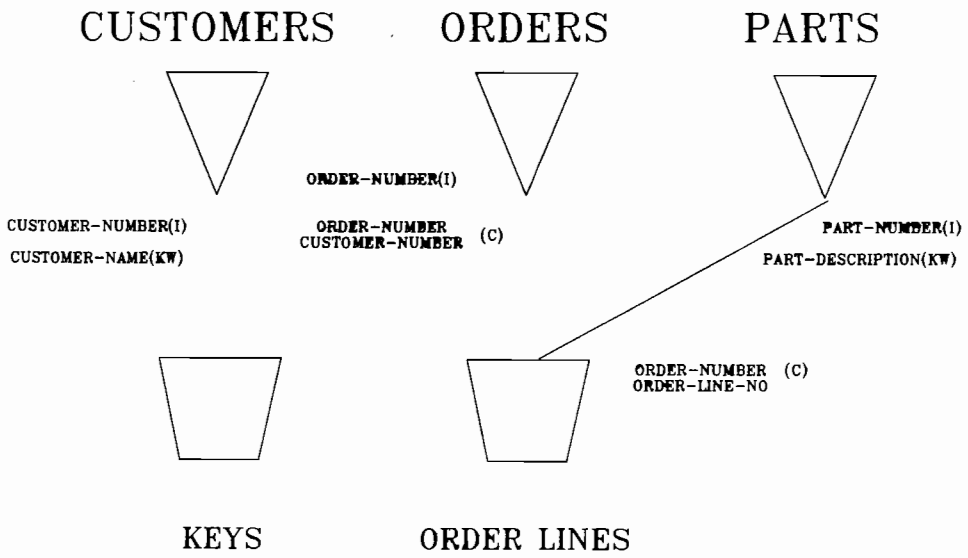
CUST.NO [_____] NAME [_____]
ADDR [_____]
ADDR [_____]
CITY [_____] ST [__] ZIP [_____]

ORDER NUMBER	ORDER AMOUNT	SHIP DATE
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]

6003-14

CUSTOMER ORDER DATABASE

KEYED ACCESS



6003-15

CUSTOMER ORDER SCREEN

KEYED ACCESS

CUSTOMER: NAME [_____]
ADDR [_____]
ADDR [_____]
CITY [_____] ST [__] ZIP [_____]

ORDER NUMBER	ORDER AMOUNT	SHIP DATE
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]
[_____]	[_____]	[_____]

6003-16

Measuring Effectiveness of Data Processing Departments

Asher F. Tunik
Computer Task Group
One Commerce Center
12th and Orange Streets, Suite 1000
Wilmington, DE 19801

Data Processing departments have become an important part of most organizations worldwide. Computers change the way we conduct business. What is even more important is that change never ends. Companies are starting to view technology as a fundamental part of their business; Information Centers are being asked to move beyond service and support and begin recommending strategic uses of technology; MIS Departments come closer and closer to departments they are supporting. Amidst all these changes, Information Center managers must make decisions and then evaluate these decisions as time progresses.

Perception VS. Reality

In recent years, users have become more computer literate and have gained their own hands-on experience with technologies such as personal computers. As more and more of the Information Systems Functions are dispersed to business units, users are wielding more clout over MIS departments.

There is often a large gap between user perception of MIS and MIS' self-image. An example could be :

> According to MIS: Our computers are up 99.5% of the time. All the systems are available for use 22 hours/day, with the rest of the time spent on back-ups. We respond to user requests within 8 working days and are striving to present some kind of a solution within 3 calendar months.

According to the
User organization: MIS never does anything right. We can never get reports on time, and it takes forever to make any kind of decisions, because reports do not present the data in an obvious way, and in most cases we can accomplish the same task manually much faster than using an automated approach.

One way to cross this gap is to ask the users their opinions of the MIS department. Once the results are in, tabulate and publish them quickly and then address the weakness points.

How To Start Self-Analysis of MIS Department	
DO	**DON'T**
Select a good cross-section of users (that means outstanding, good, average and bad performers)	Interview only outstanding achievers.
Survey company officers, middle management and first line supervisors.	Survey only one level of management.
Ask questions about overall level of satisfaction, keeping in mind functionality, reliability, and quality of services indicated	Ask questions that users cannot relate to or do not understand. Use a lot of computer jargon to impress users with your technical qualifications.
Ask users what are the most critical systems in place today. Should they be changed to deliver needed information tomorrow and what the changes should be.	Tell users what systems they should rely upon to make decisions and force them to use these systems.

Efficient vs. Effective

The Living Webster Dictionary of the English Language gives the following definition to the words Efficiency, Efficient and Effective:

Efficiency - Competence for one's duties; power of producing intended effect in relation to cost in time, money, and energy; the ratio of resulting useful work to the energy extended.

Efficient - Acting or functioning competently; able to be used with satisfaction and economy; competent; causing desired effects.

Effective - Serving to effect the purpose; producing the intended or expected result.

Although these definitions are very close to each other, they imply that efficient is not necessarily effective. Effective implies that all the actions serve the common goal or purpose. Efficient implies that all the actions are done competently and economically, but they do not necessarily fit the common goal(s) and/or objective(s).

MIS Career opportunities Broaden

Data Processing departments have been viewed as deliverers of technology; people who set users up with a hardware and a software and show them how to use it. A few things have happened since then: Users grew more sophisticated, business needs changed, and technology matured. As a result, data processing departments are getting more and more involved with recommending new uses of the technology, rather then simply supporting the current hardware and software.

As companies are coming to view technology as a fundamental part of their business, they start hiring MIS professionals to work as a part of the end-user departments. This trend is most common in information-sensitive businesses and businesses that involve high-volume transaction processing, such as banking, brokerage, insurance and retail. End-user departments look for a slightly different MIS candidate to fill positions in their departments; an employee who understands that business needs should drive the technical reality and who has a better grasp of business workings. These individuals understand not only business applications but also business implications and are more adept at supporting business needs.

Many Data Processing department members do not have the background and/or experience necessary to apply the technology to the business needs. Recognizing this more and more, MIS managers are getting involved with different departments within their companies, getting a first-hand look at the issues facing their organizations, and trying to identify and understand

trends. More and more we see a trend where departments that used to act as firefighters of sorts are now tackling the problems at hand, looking at a bigger picture, trying to identify trends across departments, and then recommending appropriate solutions. IT IS NOT ENOUGH ANY LONGER TO STAY EFFICIENT AND KEEP CLOSING REQUESTS FROM THE USER BACKLOG LISTS, MIS DEPARTMENTS NEED ALSO TO BECOME EFFECTIVE IN TACKLING ISSUES THAT THEIR ORGANIZATIONS FACE.

A case in point is an organization that I was working with a couple of years ago. Users were defecting mainframe-based applications and writing their own little system using PC software. The Data Processing department had two choices: resist or help. Resistance worked fine ... on paper. Company management issued a memo which users appropriately threw into garbage baskets and war was declared!!! The user community was still using small, numerous PC systems and Data Processing was stuck. Then the MIS people got their "thinking caps" on and decided to cooperate with the users. It took a few monthly meetings to figure out the trends and reasons why users were defecting mainframe applications and/or sabotaging them. After a few heated sessions the Data Processing department was able to adopt a new action plan and formulate a Mission statement. The system was partly modified and brought up to date, different departmental LANs were installed with departmental applications exchanging data with the centralized system daily and, at least as far as I know, things have calmed down. Company officers get up-to-date reports, which allows them to make knowledge-based decisions, and the user community has systems that "feel right" to use.

MIS Professionals need more "People" skills

What would you do if you found yourself as a part of an executive team that must resolve several critical issues in order for a company to stay competitive and ultimately to survive. The problem is: A billing cycle must be shortened and Information Systems must be able to provide up-to-date information and reporting to the management in order to identify trends and issues facing the organization.

Initially, team members will need to express themselves in their own long-winded style, and after everyone has had their moment (more like an hour), there will be a need to establish an effective communication in the group.

Each member of the team is influenced by loyalty to his/hers department needs and probably will try to protect the interests of their departments while they are pressured to develop an outstandingly brilliant plan for improvement. Your position, however, will be even more complicated. As a representative of the MIS department, you not only need to understand the issues involved, you also need to understand what the team members are saying (which might be different from what they mean); and when ways for improvement are found, you will have to make sure that your department can effectively implement them. So in order to be EFFECTIVE, you will need to listen, objectively evaluate, and then formulate clear recommendations.

The days when programmers were put in a corner with a terminal are gone forever. Now MIS staff are becoming more and more a part of the organizations responsible for making decisions and interacting with the rest of employees.

Increasing Effective Utilization of MIS

Having applications developers and/or technical support staff working side by side with end users makes users feel more involved and more likely to feel that they own the system. Through the pride of ownership, they are more likely to use the system strategically, which results in a higher payoff to the company.

DOS AND DON'TS OF SUPPORTING END USERS

DO	DON'T
Use Prototyping when developing Applications.	Keep prototyping forever - get the right applications to the right user in as timely manner as possible.
Involve end users at every step from needs assessment through post-implementation support.	Mislead users or be less than honest with them about the application you are able to provide them.
Anticipate user needs.	Sit back and wait for user requests.
Use CASE techniques and/or CASE tools to improve the development group's responsiveness to user requests.	Develop applications that do not satisfy user needs or arrive too late.
Try to identify user's concerns and trends in the organization.	Establish a wall of separation between MIS and the users.

Increasing Strategic Use of MIS

An increasing number of companies are starting to use computer technology to derive strategic decisions, which affect the course of action for months and years to come. Answers to the following questions will help to increase strategic use of MIS and Computer technology in your organization:

- o How should we be using computers today and in the years to come? How should we spend Information Systems budget dollars today?
- o What applications will remedy pressing problems? What applications will establish a basis for subsequent systems development?
- o What applications will provide immediate savings? What are the magnitudes of the savings?
- o What applications will provide long-term return on investment?
- o What applications can be achieved with existing resources?
- o What applications will create visibility for the MIS department?
- o What applications will best demonstrate the MIS department's contributions to the organization?
- o What applications will help our organization carry out its strategy and achieve its goals?
- o What applications will help our organization to create new strategy and set new goals?

Earlier, we talked about surveying users. Answers to the above questions will help the MIS departments evaluate the Efforts Spent vs. Goals of the whole organization. Also these answers will enable MIS to evaluate the current Mission Statement (or to create one) and/or update it to reflect goals of the organization.

How do you measure?

What are the symptoms of your performance, how do you measure up right now? After all, your department might be doing an outstanding job providing your organization with strategic data and helping company management identify trends and issues. Here are a few questions that will help you assess the situation:

- o Who in the organization has the computing power on their desks?
- o Do people who have the computing power use it and to what extent?
- o Who has access to the data (in the form of reports or raw data downloaded into PC's for use with LOTUS or DBASE etc.)
- o Are executive summary reports longer than 7-10 pages? If they are, probably nobody ever reads them.
- o What reports that you produce are actually being used? What can you do to make these reports more useful, if anything?
- o Do your users know your MIS people?
- o Do your MIS people know their users?
- o Do your MIS people have a feeling of ownership of the systems they work with?
- o Do your users have a feeling of ownership of the systems they work with?
- o What can you provide your users that will make their life easier?
- o What can you provide to the company management to make their decision process easier?
- o Do both the user community and their management derive value from using the systems in place?

Answers to these questions will provide you with insights on how your MIS department is doing in your organization. Once you start thinking about effectiveness, there will be more questions which you will need answers for. Some of the answers you will like, and some of the answers you will not. But it is better to identify the problems so you can address them, rather than someone else identifying them later.

Conclusion

Unfortunately, there is no "ready made" formula that will enable you to conclude whether your MIS department is effective or not. However, in order to provide your organization with the level of service that it needs, MIS should be both Efficient and Effective. A short backlog of user requests is no longer a measure of effectiveness of MIS; MIS should also be evaluated on quality of the data that is delivered to the users and ability to manipulate the data to make strategic decisions.

Acknowledgments

I am grateful to James A. Giffing and George Treisner for insights offered during the preparation of this presentation.

Bibliography:

1. Furger, Roberta, "Taking The Next Step", INFOWORLD August 28, 1989, pp 43-46.

2. Grove, Andrew, "Stuck in the middle", MIS Week, October 19, 1989, p. 36.

3. Saks, Shirin, "Career Opportunities Broaden as MIS Moves out to Departments", MIS Week, September 4, 1989 pp, 30-31.

4. Goff, Leslie, "Decentralize to Support Department End Users", MIS Week, October 16, 1989, pp 1, 37.

5. House, William C. and Napier, Herman S., "Computers in the Executive Suite: Acceptance Is Slow", The Office, October, 1989, pp 28-32.

6. Lederer, Albert L and Sethi, Vijay, "Pitfalls in Planning", Datamation, June 1, 1989, pp 59-62.

7. Goff, Leslie and Puttre, Michael, "Business Strategies Misaligned with MIS Resources"m MIS Week, November 6, 1989, pp 38, 40.

8. Moad, Jeff, "Asking Users to Judge IS", Datamation, November 1, 1989, pp. 93-98.

9. Bohlin, Ron and Hoeing, Christopher, "Wringing Value From Old Systems", Datamation, August 15, 1989, pp 57-60.

Programming Standards - Help or Hindrance ?

Patrick A. Lockwood

Orion Systems Technology, Inc.
325 South El Dorado Suite 102
Mesa, AZ 85202

(312) 894-6983

In my career in this profession, I've seen many kinds of standards in shops that program in nearly any language imaginable. Some of these were not truly contributing toward improved productivity and the quality of the final systems developed. So why am I so happy working in a shop with standards?

The answer is quite simple; good standards make my job easier, more exciting, and let me concentrate on the real problem to be solved. This probably doesn't fit with some people's picture of a bureaucratic organization with an enforcer looking over every programmer's shoulder, but in our shop it's true and it works.

Why did we implement standards that let us work in a programmer's Eden? The answer, again, is quite simple; to be more profitable. This is manifested both in direct productivity gains, and in indirect ways such as reduced maintenance due to errors, reduced training costs, and reduced turnover. As managers, profitability must be one of our goals, whether stated as increased revenue as a result of our systems, or as reduced expenses in our department or the departments we service.

Before we look at the types of standards we've implemented, let's look at what we didn't implement... and why.

We DO NOT use standards that measure productivity in lines of code per day, week, or month. I've seen this type measurement, and I can tell you that the results were the same in all the shops in which it was utilized. Programmers learned to write long programs with lots of lines. That's all they accomplished, and were unhappy doing it. And the companies spent money trying to record and measure performance against the standards.

We DO NOT use standards that measure time spent to achieve arbitrarily imposed target dates. This is a thorny problem in many shops; the common scenario for this is that the VP of Indefinite Objectives needs a new system, the Reluctant Planning Committee gives its approval, and a target date is established. Then the MIS Director assigns the project, it gets in the queue, finally gets started with its first revised target date, and thereafter the poor programmers get beat up for not meeting the second, third, fourth and Absolutely Final Target Date. You might have to make excuses to upper management, but don't use this to measure your programmers.

I've worked where this is fairly common, and my only suggestion for managers who cannot change this pattern, is to get out while you have your health. However, there are ways to alleviate some of this, one of which is to develop a good project development methodology that displays to upper management your professionalism and that of your staff. This implies a technique that helps you establish and meet reasonable goals.

And that's where standards come in.

We have developed our standards over a period of time, and they are not inflexible. We believe they save money, for us and for our end users, and have allowed us to achieve a very high level of quality in the systems installed.

They can be broadly categorized as follows:

DEVELOPMENT METHODOLGY	This is the series of steps involved in taking a project from initial vague request to an implemented, running system.
PROGRAMMING TECHNIQUES	These are the conventions we use for the actual coding of programs.
DOCUMENTATION	Yes, we actually do some. Our primary concern is with user documentation, with very little formal system documentation. Later we'll see why this is true.
TESTING	This is broadly defined in our development methodology, but we have certain common techniques we follow.
OPERATIONS	This can vary extremely, since a programmer must meet the requirements that may be imposed by another department. However, within these restrictions we have common techniques for making the job of the operations staff a little easier.

Relative to an actual program, we have established our priorities in the following sequence. First, it must work. Second, it must be maintainable. And third, it must be reasonably efficient.

Our project development methodology isn't particularly original, but it works. We follow the following stages in new projects.

I GENERAL REQUIREMENTS
This includes development of the project scope, which often also specifically states what is not to be included. The general requirements of the new system are defined, and an analysis of the current system is conducted.

II DETAILED REQUIREMENTS/GENERAL DESIGN
The general requirements are expanded into sufficient detail and documented. Alternative design concepts are analyzed, and a general system design is chosen, with the system broken down into its major subsystems. This also includes the structure of the data base(s) and key files. We also use a matrix to depict the system's ability to satisfy the detailed requirements.

At this point, we make a rough time estimate for the total project, and a detailed estimate for the next stage. This estimating technique is then repeated for each subsequent stage in the project.

III SUBSYSTEM DESIGN
The defined subsystems are broken down into their computer programs and administrative procedures. This also includes screen design, forms design, and report layouts. This normally is depicted in a hierarchical chart of the major functions and their sub-functions, and looks like a company's organization chart. On line activity is shown separately from those functions that will be done in batch.

NOTE that the following two stages are usually carried out simultaneously, depending on the size of the project and staff assigned.

IV PROCEDURE DESIGN
The required user procedures are defined, along with user training objectives and methods to be used.

V PROGRAM DESIGN
The required programs/modules are defined in sufficient detail for coding. The data base and file layouts are finalized, and program level test plans are developed. This is also where system dependent copy library members are developed, as well as specifics for system dependent callable utility modules.

As before, a revised total project time estimate is prepared, along with a detailed estimate of the next two stages. These next two stages may also be done simultaneously.

VI PROCEDURE DEVELOPMENT
The required user procedures are prepared, as well as a final training plan. For user procedures we like to follow a common format that relates as much as possible to the on line screens the users will see.

VII PROGRAM DEVELOPMENT
Finally, someone gets to code. By this time we have a lot of documentation developed in the earlier stages, so we aren't worried about doing a lot of 'system documentation'. We use our design documents as system documentation.

We also don't do a lot of individual program documentation. Our requirements and user documentation define screen input format, editing rules, and processing, as well as the functions and input/output formats for batch programs.

Our normal programming technique is to constantly borrow from existing programs, following the old adage of 'Programmers Wise Plagiarize, Let No Other's Work Evade Your Eyes'.

After all, how much different can a Customer Maintenance program be from a Vendor Maintenance program. The basic functions are the same; add, change, show, or delete. So the basic program structure is the same, with the only differences being the screen formats/edits and the data base/file layouts.

We use so many common copy library and utility modules, that this type of program is quick and easy.

Simple reports also follow patterns; select, sequence, and list (with or without details and/or various levels of totals).

The individual programs are unit tested and a system test plan is finalized. This includes test cases, as needed, to ensure that the interfaces between components and other systems are proper.

VIII SYSTEM TEST
Test data is prepared as needed, and all components of the system are tested to ensure they function individually and when combined. User procedures are also tested to ensure that the computer programs and procedures together actually meet the original requirements.

Interfaces to other systems are tested, and if needed, volume and/or stress testing is done to validate the system's ability to meet any throughput and response time requirements.

Where needed, test results are predicted PRIOR to actual testing of system functions. We've learned that programmers test that which already works, so we do require that certain system results be predicted in advance and that the tests match the anticipated results.

Final quality assurance plans are developed, and usually take the form of parallel operations.

Final training plans are developed, and the intitial sessions scheduled.

IX IMPLEMENTATION
The quality assurance testing is completed, and if needed, the system is revised.

Formal training is begun. It may include both classroom and 'hands on' sessions. To the extent possible, we like to use user supervisory personnel for most of this.

Any required final conversions and/or startup data entry are done and the system becomes the property of the users.

NOTE that all of this implies that user personnel are actively involved throughout the course of the project. We also like to involve the company's Internal Audit department throughout the project. They sometimes tend to be a little bit of a pain, but their cooperation early in the requirements and design stages can save you many later headaches.

It also looks like a pretty formidable way to go about a new project. The real truth is that the level of formality in all stages is directly related to the size of the project. And, in some cases, the stages tend to overlap a little.

Finally, the programming effort is depicted as if almost negligible. Of course, this is far from the truth. But the effort required for many programs in a large system can be minimized by following standardized programming techniques. This is exactly what the 4GLs do, but if you don't use one, you can duplicate many of their advantages by establishing common program structures and routines. This gives you both development speed, and reduced testing requirements.

We are primarily a COBOL shop. Some of the techniques we and/or other shops use are:

COPY LIBRARIES

We use two kinds. One is for commonly used Working Storage, Linkage, and Procedures. Representative entries are such things as IMAGE and VPLUS calling parameters, common IMAGE and VPLUS routines, and the Working Storage and procedures for frequently used functions such as centering a line of text, date conversions, etc.

The other type of copy library is for system dependent data such as file/data set definitions and system dependent routines such as the calculation of an A/R due date based upon terms and invoice date.

Our basic approach to copy libraries is: 'If it's used more than once, put it in a library'.

MACROS

If you're not familiar with the COBOL macro facility, investigate it. They seem to be a little harder for newer programmers to understand and work with, but they can reduce coding effort.

To us, the major difference between macro functions and copy library routines is that macros insert in line code, while copy library procedures tend to be performable.

The two may be used in combination.

CALLABLE FUNCTIONS

Sometimes its just downright inefficient to include a major routine as a copy member in every program that needs it. Dynamically callable subprograms can be utilized to provide a common procedure.

These may be either common or system dependent.

These three techniques may not be considered standards by some, but to us they are in the sense that they easily allow programmers to follow a consistent pattern. They take a lot of the drudgery out of coding, allow the programmer to concentrate on solving the real technical problems, and create programs that contain significant amounts of proven logic.

These next pages discuss some standards we use that are highly subjective. However, we strongly recommend you adopt similar ones. Working in a shop that does some work for client companies has exposed me to more variations in shop standards than just those used in shops where I've been employed, and I've seen the results of haphazard programming standards.

PROGRAM NAMING

We use mnemonic names that follow patterns. For example: Customer Maintenance is CUSTMNT, Terms Maintenance is TERMMNT; Customer Report is CUSTRPT, Terms Report is TERMMNT. This makes it easy to edit the CUSTMNT source file and change 'CUST' to 'TERM' in all, and be well on our way towards creating a new maintenance program.

We also designate Source, USL, Job, and Executable files by simply adding the proper suffix. This creates a set such as CUSTRPTS, CUSTRPTU, CUSTRPTJ, and CUSTRPTX; all readily identifiable as CUSTRPT related files.

GROUP/ACCOUNT NAMING

Within each development account, we maintain separate groups that also follow a pattern. Each major system is assigned a two character designator, such as 'AP', 'AR', or 'GL'. This system designator is then used as a prefix for naming groups as follows, where 'xx' is the system designator:

xxSOURCE	All source files, including system dependent copy libraries, $INCLUDE, or macro files.
xxUSL	All compiled USL files.
xxJOB	All job stream files for compiling, linking and testing programs.
xxDOC	All HP3000 resident documentation files.
xxFORMF	All VPLUS forms files. We name the original forms file xxFORMS and the compiled (fast) version xxFORMF.
xxTEST	All production files remain in the above groups, while all files related to the current testing are moved into the xxTEST group until testing is complete. Only then are they moved into the production groups.

I've seen variations on this theme, many of which were good. For example, one major application software supplier carries a similar technique even further by including the current major release version as part of the group name. (AP3SRCE is source code for version 3 of AP, AP3USL contains version 3 USLs, etc.).

We NEVER have source code or any other development files in the accounts where production programs reside or users log on. When all testing is complete, executable program files, VPLUS Fast forms files, and any required production job files are copied to the proper production account.

We like to put production job files in a production group that has restricted access, thereby eliminating the security breach that Read access allows.

We prefer to have no production job files as part of user applications, both to eliminate potential security problems and to avoid the chance that they will be purged and/or modified to prevent their proper execution. Where possible, we have any required batch processing intitiated from within the on line system by dynamically building, streaming, and then purging job files.

As part of our effort to make our programs look like they all had the same author, we've adopted some standards for use within programs.

PARAGRAPH NAMES

Many people like numbering to indicate where, or at what level a performable routine can be found. If you like them, use them, but keep the numbering scheme consistent.

In all cases, paragraph names should be long enough and clear enough to indicate their real function.

Some of our paragraph names are mandatory for certain functions such as END-OF-PROGRAM and DB-ERROR-EXIT, since they may be referred to by common copy library routines.

Others are mandatory to maintain consistency between programs, for example in working with VPLUS screens we always have the following paragraphs where 'scrn' is the actual form name within the forms file:

scrn-SCREEN; the paragraph that sets up the form name and processing control to retrieve the form via a common copy library routine.

scrn-INIT; the paragraph that intializes the buffer and window message when first entering the screen. Then the form is shown on the terminal by a common routine.

scrn-KEYS; the routine that reads the terminal and determines what to do if one of the function keys was pressed.

Continued...

scrn-EDIT; the routine that edits the current form, and causes it to be re-displayed if errors are found.

scrn-FINISH; the routine that is used after all edits have been passed. Typically, this is where the data base is updated in a standard maintenance program.

These kinds of naming conventions make it very easy to both copy and modify programs, but also for a programmer to find the proper paragraph in a program that she has never been into.

Like techniques are used for other programs or parts of programs that lend themselves to standardized naming. In our earlier discussion of changing 'x' to 'y' in all lines of a source file, we only looked at the program name. You can easily see how this may be used for things such as screen names, data set names, etc.

These methods turn trusty old COBOL into a 3 1/2 GL.

Some other standards we use have to do with actual program structure. We're not died in the wool structured programming freaks, but we don't like spaghetti code, and we don't like hard to read programs.

SPACING

Use $PAGE with headings to identify major routines and working storage sections. Use space between lines and within lines.

The paper is cheap compared to the time it takes to find something in the program.

GO TOs

Yes, we allow them, but only where appropriate. GO TO xx-ERROR-EXIT does not destroy a well structured program, and can save a lot of repetitive switch testing.

We also use them in loop constructs.

We seldom use GO TO routine-EXIT, but it is acceptable.It's commonly used with perform xxx THRU xxx-EXIT, often with intermediate paragraph names. We've uncovered some sneaky bugs as a result of a programmer inserting a routine somewhere in the middle of such a format.

Continued...

NESTED IF/ELSE Set a limit! Ours is three, anything required that won't fit in three is handled by a performable routine. For example:

```
IF A = TRUE

   IF A1 = TRUE

      PEFORM A1-ROUTINE

   ELSE,

      PERFORM A2-ROUTINE

ELSE

   PERFORM B-ROUTINE.
```

SECTIONS On classic HP3000 machines, sections with the different numbers create code segments, and that means swapping.

VERSION NUMBERS We actually have them in Working Storage so that they be shown on screens.

We include comments that contain the description of changes made to the current version.

In summary, standards are meant to save time and money, as well as produce good systems. My own experience has been that standards only work when the people using them see some benefit, and when it's easier to follow the standards than it is to avoid them.

Use your experience and that of some of your better staff members to help you develop your own shop standards, then make the standards and their documentation part of your in house training program.

You'll be glad you did.

STANDARDS

Why ?

Development

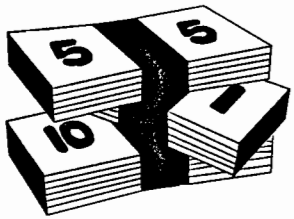
- Utilize 'Skeletons'
  - * Maintenance
  - * Maintenance Lists
  - * Reports
  - * Batch Updates
  - * Other Functions
- Copy Forms Files
- Common Routines

Maintenance

- Author 'Anonymous'
- Common Techniques
- Common Naming

Documentation

- Utilize 'Skeleton'



Why ? To save money and create better products

STANDARDS

Make it easier to follow standards than to avoid them.

Copy Libraries	Common library for commonly used Working Stoarge, Linkage, and Procedures. Application specific libraries for file/data base items, application specific Working Storage, Linkage, and Procedures. Standardize naming conventions for copy members and paragraph names within them (xxxxxxW, xxxxxxxP).
MACROs	Standardize naming conventions. (xxxxxxxM).

MACROs generate in line code, procedure copy members require PERFORMs.

SLs & RLs	Restrict assess to changes to sub-programs that will reside in SLs or RLs. Use for "utility" routines.
Program Naming	Use mnemonic names, follow a pattern. (LOCMNT, LOCRPT). Use prefix or suffix to designate type (LOCMNTS, LOCMNTU), or separate groups (xxSOURCE, xxUSL), or both.

6006-13

STANDARDS

JOB File Naming

If you must have permanent job files, use a naming convention. Store them in separate, restricted access groups (xxxxxxxJ.JOB).

Passed Parameters

Use standard copy member to carry information used by all programs. Include a STD-CALL-USER-PARMS for program specific parameters. This also makes it easier to dynamically stream programs with these parameters passed in INFO = or ACCEPT lines.
Establish copy members for terminal and data base parameters.

Utility Documentation

Prepare a simple manual documenting all RL, SL, and performable copy members and the working storage copy members/$INCLUDE files required for each. For example:

```
CENLINEP in STDLIB.

Requires CENLINEW in STDLIB.

Centers TEXT-IN in TEXT-OUT for LINE-LENGTH characters.

MOVE STD-CALL-LOC-NAME      TO TEXT-IN.
MOVE 30                     TO LINE-LENGTH.
PERFORM CENLINEP.
MOVE TEXT-OUT               TO STD-HEAD-LOC-NAME.
```

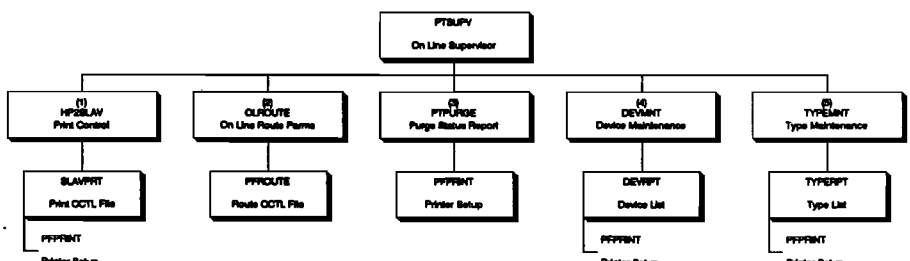
STANDARDS

System Documentation

If you never do anything else, prepare a chart showing the hierarchy of all programs initiated from within the application.

STARPRINT

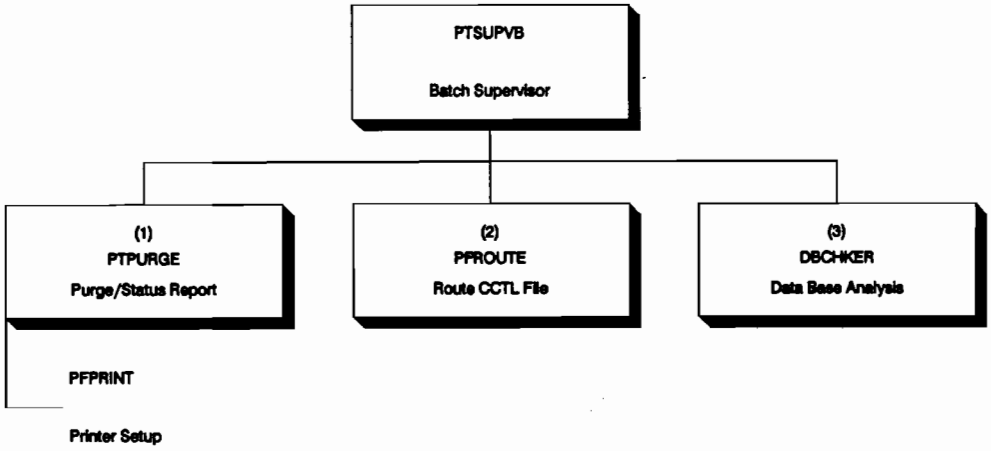
Program Organization



NOTE that the numbers in parenthesis indicate the main menu (PTSUPV) selection numbers.

STARPRINT

Program Organization



NOTE that the numbers in parenthesis indicate the "ROUTINE-CODE" passed as part of INFO = "parms".

STANDARDS

Paragraph Names	If you like numbering, use it. We don't; our experience shows it detracts from the ability to read the names. In any case, make them meaningful.
Data Names	Again, meaningful. Some shops like prefixes such as "DB-", WS-", and "xxx- (xxx- being an abbreviation for a record name)". Many of these conventions can be established in your copy libraries.
Paging/Spacing	Use $PAGE "*routine name*". We like "Continued next page" when a single performable routine exceeds one page. Paper is cheap compared to a programmer's time. Use plenty of white space; line up PIC, TO, etc.
GO TOs	Avoid, but use where appropriate. We use GO TO xx-ERROR-EXIT, and have no problem following program logic. We also use GO TO *start-of-routine* in looping constructs. Some on line programs become more complicated when processing is heavily function key driven. GO TO *routine-exit* is acceptable, but we also avoid that.
Nested IFs/ELSEs	Set a limit and stick to it; use AND/OR and additional PERFORMs. If you prepare pseudo code for programmers, name routines for them. Align IF and ELSE statements with matching indentation.

STANDARDS

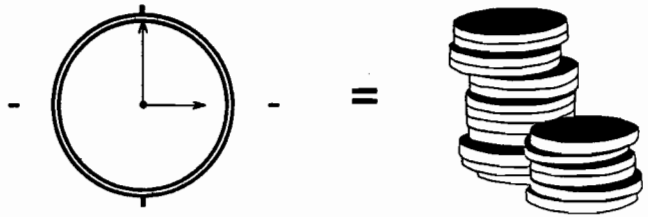
PERFORM THRU	Avoid; you're just asking for someone to insert a routine where it doesn't belong. PERFQRM THRU *routine*-EXIT is O.K.; we generally even avoid that.
SECTIONS	Use only if needed to segment program or if required by COBOL rules (SORT, DECLARATIVES). **DO NOT** put all your I/O copy members in a special section. If you must segment, put housekeeping, error handling, and end of program routines in the same segment. Don't put loop controls in a different segment than the loop they are controlling. If you must segment, use the program name as the first 8 characters of the section name to avoid segment name duplication for combining multiple sub-programs with one main program in the SEGMENTER.
Comments	Describe the program's function on the first page; it helps to describe why any files/data sets are used. Don't waste other people's time with meaningless comments. If the program is well written, it shouldn't need comments in the body of the program except for tricky routines.
Skeletons/Models	Build them and use them for maintenance programs, reports, updates.

STANDARDS

Clean Up !

Programmers wise plagiarize, but many forget to clean up after their hack of program *X* into program *Y*. Excess code means excess time for the next modification.

Remember that standards should be used to help you save valuable time and create better products. This applies for the entire spectrum of design and development.



USER DOCUMENTATION

Yes ! Do it; if you've done your preliminary design work well, it's fairly easy.... even if not very exciting.

System Objectives	Briefly describe what the system is designed to accomplish.	
User Interface	Describe system conventions for on line processing; such things as screen handling (block mode/character mode), how errors are handled, common function key usage, date and amount field entry.	
System Overview	Describe how the system meets its objectives. Include an overview of processing and timing of required functions.	
Screen Descriptions	Purpose	Describe what the screen will accomplish.
	Security	Who is allowed to do what.
	Helpful Hints	Hard to define, but many screens could use them.
	Warnings	That which you may not want do, but is allowed.
	Function Key Usage	What happens if you press one.
	Field Descriptions	Allowable entries and their meanings, defaults if not shown, format, and whether the field is required.

Put as much help/description as possible on the screen itself, including interpretations of entered values, such as Vendor Name, Account Description, etc.

DISTRIBUTING SYSTEMS ACROSS MULTIPLE COMPUTERS - IS IT EASILY ACHIEVABLE ON HP3000S?

David B. Wiseman
Proactive Systems Inc.
339 South San Antonio
Los Altos
CA 94022
(Tel: 415-941-9316)

ABSTRACT

The author will review the problems that arise when distributing applications and data across multiple HP3000s.

These problems fall into managerial and technical categories. The author will relate his own experience in building such networks and how the problems were overcome.

The concepts and requirements of distributed data base software will be discussed and the possible solutions based upon IMAGE, HP SQL and other DBMSs will be outlined.

THE BENEFITS OF DISTRIBUTED SYSTEMS

Distributed computer systems are a hot topic at present because many companies recognize that they can build systems that match more closely their organizations needs and at lower cost using distributed data base technology.

Distributed systems simply use multiple computers to disperse processing in a network instead of using one centralized computer. They range from the extremes of two computers placed back to back to large networks of linked computers spread internationally. The benefits of distributed systems are:

- You can minimize computer hardware and data communication cost. Multiple smaller computers give you more effective power for your dollar than one large computer.

- You can minimize data communication costs. Doing local processing minimizes data transmission needs.

- Increased data redundancy can provide 24-hour uptime, full disaster protection (you don't have all your eggs in one basket) and better user response times.

- Geographically dispersed organizations can obtain local processing and local management control.

- Computing capacity growth can be more incremental - simply add another computer to expand your processing capability.

- There is more flexibility in hardware selection, computer placement, management structures, application design, etc.

There is an optimum mid-point between a totally centralized approach and a totally decentralized approach - see Figure 1.

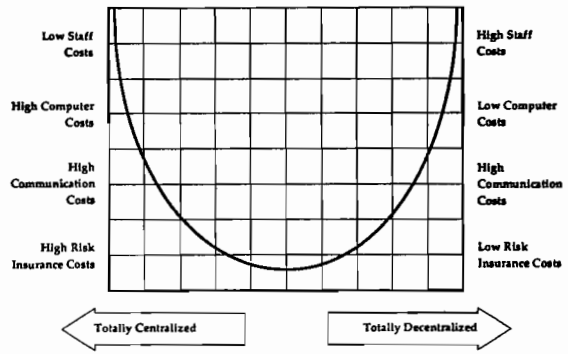


Figure 1 - Overall System Cost

However, traditionally the construction of distributed systems has raised a number of problems, both technical and managerial. Below I'll go through the nature of these problems and then tell you about the solutions that are available on HP3000 computers.

DATA IN THE WRONG PLACE

Once you spread your data bases over multiple computers, you need to be able to:

- Easily access data on remote nodes with no more difficulty than if it was on the local node.

- The system or the user, but preferably the former, must know where the data is located.

Both of these can be summarised by the phrase "data location transparency". Users or application programs should not need to have to worry where the data is located. See Figure 2.

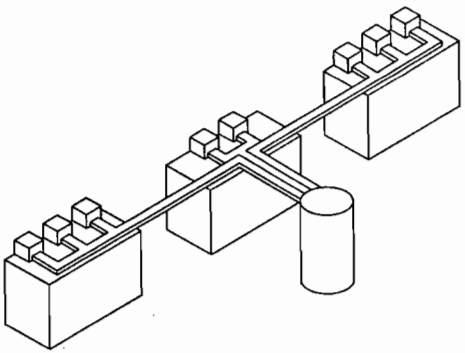


Figure 2 - Data Location Transparency Example

POOR PERFORMANCE

All of you who have used NS or DS know that the speed of access to remote data bases, both in terms of response times and data throughput, can be apparently much worse than local access. If you have a lot of user sessions accessing remote IMAGE data bases, your whole system can grind to a halt.

EXPENSIVE DATA COMMUNICATIONS

Remote data base access using standard NS facilities not only uses more cpu processing capacity but also generates a lot of data communications traffic - this by its very nature costs more than a local disc access. Data communication costs can be a significant proportion of overall project costs when building distributed systems.

RECOVERY ON FAILURE AND MAINTAINING TRANSACTION INTEGRITY

If a network failure occurs then data recovery can be a problem. Whereas IMAGE provides data recovery capabilities on a local update to a single data base (e.g. transaction roll-back), it does not provide recovery of transactions that span multiple data bases, some of which may be on remote nodes - this is common need in a distributed system. Relational data bases typically have similar limitations unless additional facilities have been added. Take an example. Say we have two copies of a data base on two separate nodes, one local and one remote. If we identically update both copies sequentially (e.g. update local copy and then update remote copy), how do we recover if the link fails after the local copy has been updated?

DATA CONSISTENCY AND CONTROLLING CONCURRENCY

Maintaining logical data consistency in an application is more difficult in a distributed system than in a centralized, one data base, approach. For example a common requirement is to share a replicated copy of a data file (see Figure 3).

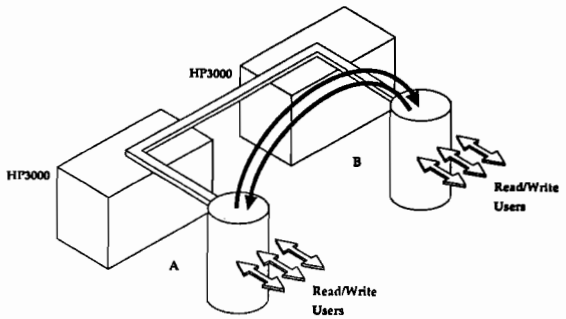


Figure 3 - Data Replication

What happens if independant users on the two computers both update the record at the same time? You either have to cope with this situation or design it out.

SENSITIVITY TO NETWORK FAILURE

If you have a large network then the chance of one or more nodes or data communication links being out of action at any one time is high. The increase in possible failure points causes a geometric increase in the overall percentage failure probability. Therefore distributed systems must be designed with such risks in mind. On the other hand distributed systems are much less likely to suffer total failure as with a centralized system - it's no longer an all or nothing scenario.

MANAGEMENT CONTROL

Distributed systems, at least ones that are geographically dispersed, tend to require different management structures to centralized systems. Staff control is more problematic and building centers of expertise more difficult. The ethos and policy of the organization is also important. In my experience there is little point in trying to install a totally decentralized computer system in a company where all decision making and management control is traditionally totally centralized - the resulting conflict in management styles will lead to disaster. In practice you tend to always need centralized management information even if much of the routine transaction processing is decentralized so computer system network and application design must cope with that.

NETWORK DYNAMISM

Anyone who has built a distributed system knows that the hardware never stays the same for very long. A strength of distributed systems is that you can build your processing capability in an ad-hoc manner (have you run out of power on two computers? - answer: simply buy a third). However you therefore need to be able to easily reconfigure data base access, processing location, etc.

THE SOLUTIONS

Solutions to the above problems fall into three categories: technical (eg. software systems and supporting tools), application design and managerial.

They vary depending on whether you are intending to use network data bases such as IMAGE, or relational data bases such as HP-SQL. Each of the above problems is covered in turn below.

SOLUTIONS (TO DATA IN THE WRONG PLACE)

On HP3000 networks, HP provides NS as a partial solution - for example it provides access to remote IMAGE data bases via RDBA. However users or application programs need to know where the data is located (or you have to set up UDC commands to provide access which is then not easy to change). Both for HP-SQL and IMAGE data bases there are now software products available (from HP and third parties respectively) that provide much more generalized and more flexible solutions to this problem. They typically have a dictionary on each node that holds data base routing information.

Every data base open is intercepted and looked up in the dictionary to see whether local or remote access is required, and, if remote, on which node it is located. This provides true data location transparency. Users or programs don't need to know the data location - the system manager can easily re-route access if he moves a data base. In the case of the product my own company produces we've also added the following:

- Access for read-only users can be routed differently to update users - this can be useful for load spreading, e.g. routing enquiry/ reporting users to a replicated copy of a database.

- Access to an "alternate" data base is automatically provided if the primary routing does not provide a successful data base open. This can improve system resiliance.

Another solution to having data in the wrong place is simply to replicate it over each node in real time. There are several products that can do this for IMAGE data bases, but it tends not to be available yet for relational data bases. We have even built systems that effectively share parts of a data base over two systems - see Figure 4.

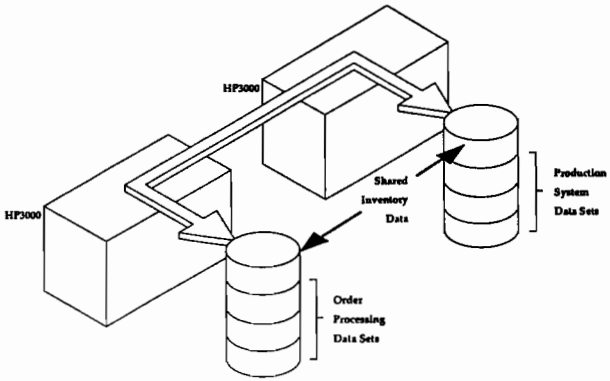


Figure 4

SOLUTIONS (TO POOR PERFORMANCE)

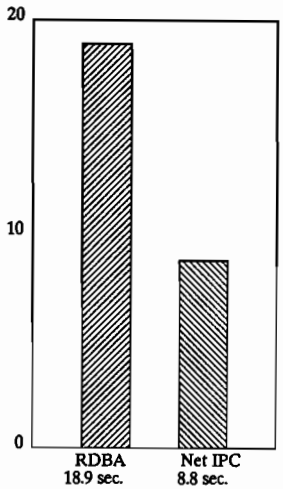
Maintaining multiple copies of data bases (or subsets thereof) can also do wonders for performance. Having a local copy of a data base can be cost effective if the ratio of read to update access is high, as is common in many commercial systems.

For example reference information such as price lists are best held on every node. The amount of disc space need to do this may be small and it reduces dependance on one central copy. Any updates (which are typically low volume to such files) can be automatically replicated from the central site - possibly in pseudo batch mode if you don't need real time updating and prefer not to keep the data comm link open.

Data replication also enables you to maximize performance by using load spreading. For example you could off-load all enquiry and reporting activity to a second copy and thus improve update response on the primary.

Another approach is to tackle the performance problem at a technical software level. Two particular performance bottlenecks that can be solved on IMAGE data bases are:

■ It takes a long time using RDBA to open a communication link and open a data base in response to ad-hoc enquiry or update activity. The solution is to hold a pipeline open to the remote data node so that response can be immediate (or at least comparable to local data base open time). Figure 5 shows some comparative figures I obtained comparing the two methods on our own linked development systems.

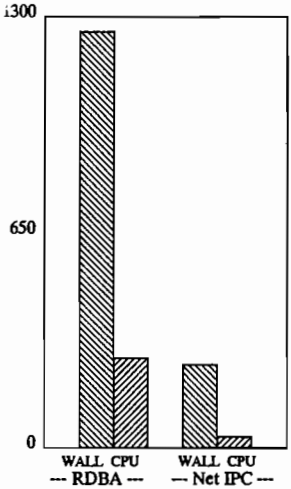


NetIPC versus RDBA

Elapsed time in seconds to establish remote session, open data base, read three records from master set, read one detail set record and close data base. Average of twelve runs accessing MPE/V system from MPE/XL system.

Figure 5

■ Poor throughput or response times due to the large amount of data transmitted by RDBA over communication links which are always relatively slow in comparison to a local access path. For example RDBA transmits the whole data base buffer which is created on a remote node. However typically you don't need all the data items in all the records. You probably only want some of the data items from some of the records. Solution: do the processing on the remote node (where it is local to the data base) and only transmit the data required by the user program. Figure 6 shows some comparative figures I obtained.



NetIPC versus RDBA

Time in seconds to perform a serial read of 7000 entries from master dataset residing on a MPE/V system from a MPE/XL system via NS.

Figure 6

Both of the above techniques can be implemented in a generalized manner and we did this using the NetIPC service of NS, ie. you don't have to change the application software to get these kind of performance improvements. By using NetIPC and appropriate buffering strategies you can effectively optimize what you get transmitted over a network which is the key to performance maximization. Making the best use of each NS transaction is the key!

SOLUTIONS (TO EXPENSIVE DATA COMMUNICATIONS)

Using the above data replication and software optimization techniques can reduce the data transmitted and hence the cost by a factor of up to 10. Also careful consideration to optimal data communication configuration (which is outside the scope of this paper) also pays good dividends.

As always, careful application design is also a key factor in minimising data traffic and maximising performance.

SOLUTIONS (TO DATA CONSISTENCY AND CONCURRENCY)

On relational data bases the solution to distributed updating is normally a 2-stage commit mechanism. This is effectively a global locking type of strategy and depends on all nodes being available plus a high transaction overhead.

On network data bases not only is it more complex to provide but performance tends to be more critical. We took an approach of using IMAGE log transactions for data replication so that we can provide recovery on a logical transaction basis. Data concurrency control in replicated data base arrangements can be handled by a variety of methods, e.g. suitable application design, providing user exception processing capabilities after a conflict event has been detected, etc.

In practice, by careful consideration these problems are not as difficult to overcome as may be imagined.

SOLUTIONS (TO SENSITIVITY TO NETWORK FAILURE)

Replication and data location transparency are the answers.

SOLUTIONS (TO MANAGEMENT CONTROL)

The design of appropriate management structures to match the needs of distributed systems are important. If you are going to geographically disperse processing responsibilities you will probably need to perform staff retraining - local management will have more responsibilities so you may need to strengthen their skills or even hire extra staff.

Sooner or later, the MIS manager tends to be running a dual organization that is divided in two ways - one to provide support for data comms and specialist software in a technical hierarchy and one that aligns with the user management structure. This dual "networked" organization structure can generate a lot of staff dissatisfaction in what was previously a strictly hierarchic organization. Often the guys at the bottom of the pyramid have dual responsibilites and no clear instructions. Moral can fall rapidly if a lot more attention is not paid to making each persons role clear to him/her.

My own feeling is that "networked" management structures take a lot of effort to set up and run effectively. The costs are often hidden however. If you can distribute the computer processing and yet retain a clear management structure with clear centers of expertise, clear career development paths and clear chains of commmunication then so much the better.

If an organization has worked well in the past, be very careful about throwing it away and starting from scratch. As always, if it works - leave it alone.

SOLUTIONS (TO NETWORK DYNAMICS)

Don't build rigid hierarchic networks and do keep your options open. Use the software and application design techniques mentioned above to retain flexibility in your systems. Don't build a rigid management structure. Try to retain staff flexibility by providing some job rotation and training your staff so that they develop a broad range of skills.

CONCLUSION

To summarise, building distributed systems is possible now on HP3000s. Using the techniques described above you can construct high performance applications where costs are lower than they would be otherwise; plus you can meet user needs more effectively.

Although building distributed systems requires learning some new tricks, the software tools are now there to help you.

REFERENCES

Brown A.J. (1989). Distributed Data Management from HP. Interex San Francisco Conference Proceedings.

Glasson L.A.R. (1989). Allbase/Net: Remote Data Base Access. Interex San Francisco Conference Proceedings.

Solland L (1989). Data Design Considerations for Distributed Applications. Interex Nashville Conference Proceedings.

Lawson R.W. (1989). Building Distributed Systems on HP3000 Computers. Proactive Systems.

Lawson R.W. (March 89). Distributed Data Bases - The State of the Art. HP Professional.

Martin J. (1981). Design and Strategy for Distributed Data Processing. Prentice Hall.

DEPARTMENTAL COMPUTING

IN A

UNIVERSITY ENVIRONMENT

PRESENTED BY: T M O'BRIEN
DIRECTOR
COMPUTER SERVICES DIVISION
UNIVERSITY OF NATAL
KING GEORGE V AVENUE
DURBAN 4001
REPUBLIC OF SOUTH AFRICA
031 - 816-2464

INTEREX-ICMS LAS VEGAS 1990

6009-0

DEPARTMENTAL COMPUTING IN A UNIVERSITY ENVIRONMENT

PRESENTED BY: T.M. O'BRIEN

INTRODUCTION TO SOUTH AFRICA

There is often some confusion as to where South Africa actually is, despite the extensive publicity the country seems to attract. Therefore let me briefly explain where it is and also describe the environment in general terms.

The continent of Africa straddles the equator with the lower area in the Southern Hemisphere.

There is an area referred to as Southern Africa which consists of the following countries: Angola, Zambia, Mozambique, Zimbabwe, Namibia (all of which are sometimes referred to as the front-line states) and the Republic of South Africa.

South Africa is the most southern country on the African continent. Within south Africa there are two completely independent land-locked countries, Lesotho and Swaziland.

The main differences between South Africa and other countries on the African continent are probably its very sophisticated infra-structure and its relative political and economic stability. Also it has one of the most diverse populations, with each group wishing to retain its own identity, which means we have many unique problems. To give some idea of the extent of this diversity the following statistics may be of interest.

Languages in South Africa are as diverse as the customs and cultures of its people. The two official languages, English and Afrikaans, are the mother tongues of 94 per cent of the white population as well as a large number of Asian and coloured people. The dominant Bantu languages are Zulu and Xhosa, spoken by more than half the black population, followed by Tswana, Northern and Southern Sotho, Tsonga, Venda, Sindebele and Swazi. The remaining languages consist of a variety of European and Indian languages and the remnants of San and Khoikhoi languages. The following tables show the individual components of South Africa's language structure and include the Republics of Transkei, Venda, Ciskei and Bophuthatswana.

Black Fifty-five per cent of the black people of South Africa speak Zulu or Xhosa. Zulu is predominant in Natal and KwaZulu, while Xhosa is the main language in Transkei, Ciskei and the eastern Cape.

Venda	3%
Tsonga/Shangaan	4%
S. Sotho	9%
N. Sotho	11%
Tswana	12%
Sindebele	3%
Swazi	3%
Xhosa	27%
Zulu	28%

White The majority of white south Africans (57 per cent) speak Afrikaans, while 37 per cent speak English. Only one per cent use both languages at home. Five per cent speak other languages.

Afrikaans & English	1%
Other	5%
English	37%
Afrikaans	57%

Coloured The home language of 88 per cent of the Republic's coloured people is Afrikaans. Eleven per cent speak only English at home, and less than one per cent speak both English and Afrikaans.

Other	0.2%
Afrikaans & English	0.8%
English	11.0%
Afrikaans	88.0%

Asian South Africa's Asian community speaks five main languages: English, Tamil, Hindi, Telegu and Gujarati. One-and-a half per cent speak Afrikaans and 11,5 per cent speak other languages.

Afrikaans	1.5%
Other	11.5%
Telegu/Gujarati	12.0%
Hindi	19.0%
Tamil	24.0%
English	32.0%

The 1980 census indicated the population as being approximately;

Black	25 000 000
White	5 000 000
Coloured	3 000 000
Asian	1 000 000
Total	33 000 000

(these figures exclude Bophuthatswana, Venda and Transkei).

The estimated population in the year 2000 will be 45 000 000 with the increase being mainly in the black component.

Apart from all the other problems that this increase poses, the one of having enough water is probably the most critical.

The bright lights have always been a lure to man; and in South Africa's complex economic and racial setup, the drift to the cities has influenced population distribution over the past two decades. South Africa's most populous city is Johannesburg, with 1,7 million people; followed by Cape Town (1,5 million), Durban (1 million), Pretoria (800 000) and Port Elizabeth (600 000).

The development of industry and commerce, and the job opportunities they offer, have been mainly responsible for the rapid increase in the urban population. Whites, particularly, have been forsaking traditional farming communities for the cities; with large-scale depopulation denuding the Karoo, southern Orange Free State and the western and northern Transvaal. People from these areas are moving in increasing numbers to Pretoria, the Witwatersrand and the Vaal Triangle.

The position for blacks has been complicated by the establishment of national and independent states, and the Government's, until recently, strict influx control and

residency rules. But for these factors, South Africa's cities would have experienced far greater increases in population. Nevertheless, projections on the future growth of our urban populations indicate that the movement of blacks to the cities will be one of the major population problems of the next century. The eight million urban blacks in South Africa in 1980 are expected to increase to at least 57 million by the year 2050. This figure compares with an estimated 6,78 million urban whites; 4,95 million urban coloured people and 1,44 million urban Asians.

This drift to the cities is already manifesting itself with the rapid increase in the establishment of 'shanty towns', the construction of which is very difficult to control. This is a problem not unique to South Africa, but being experienced by other third world countries all over the world.

What has all this to do with Departmental Computing? The diversity of population impacts everything that happens in South Africa and especially on the University campus. Computer-Aided Education and computer literacy in general are but two of the factors we must consider when introducing computer-based systems.

Personally I feel privileged to live in such a richly diverse mixture of cultures, religions and languages. South Africa is an exciting place with a tremendous future.

A BRIEF HISTORY OF THE UNIVERSITY

In the first years of this century, a technical education commission recommended that ...

> ... a University College should be established in Pietermaritzburg, with a provision for an extension of its work to meet the needs of Durban and also of other centres when the need arises.

A University college Bill was put before parliament and promulgated as Act No. 18 of 1909, and the Council, Natal University College, met for the first time on 21 January, 1910. Fifty-seven students were registered in February, and the fist classes were held in Maritzburg College. The courses led to the intermediate and final BA examinations in arts and science of the University of the Cape of Good Hope. Meanwhile, the Pietermaritzburg corporation had generously donated some 40 acres of land at Scottsville, and the first building was opened in August 1912. On 2 April 1918, the Natal University College became a constituent college of the University of South Africa.

The most important development after the first world was the extension of NUC to Durban. The Natal Technical College, founded in 1907 as the Durban Technical Institute and now the Natal Technikon, was already giving courses at university level in engineering and commerce. In 1922, NUC took these over and admitted the lecturers to Senate.

Through the generosity of a Durban man, Mr. T B Davis, who wished to honour the memory of his son, Howard, who had died in the Battle of the Somme, Howard College was built on a site on fifty acres of land in the Stella Bush granted by the Durban City Council. In August 1931, Howard College was opened and classes were transferred there from the Natal Technical College.

In 1936, the City Building (later renamed the Oldham building) was erected on Warwick Avenue to provide a centre for part-time classes.

Normal growth of the College was checked during the second world war but, in the years that followed, extensive capital development took place in both centres. A ten-year plan for development was launched to give direction to a drive for funds by the Natal University Development Foundation. In 1946 the government approved the establishment of a Faculty of Agriculture at Pietermaritzburg and, in 1947, the establishment of a Medical School for African, Indian and Coloured students in Durban. In both centres of the University the campuses were enlarged until today they comprise approximately 162 hectares in Durban and 113 hectares in Pietermaritzburg.

Independent University Status and Dual Campus Development

Because of its rapid growth in student numbers, its range of courses, and its achievements in and opportunities for research, the Natal University College was granted independent University status in 1949.

At the end of 1973, it was decided that all teaching departments in the two centres should be fully independent of one another and a number of additional appointments were made to ensure that members of the academic staff should not be required to travel between the two centres to give lectures. General administration and finance continue to be principally centered in Durban.

The number of students seeking admission to the University grows annually, and the total enrollment in 1988 was 12 724.

The University and Society

The University of Natal is committed to service through excellence in teaching and research. It is opposed to discrimination on grounds of race, colour, nationality, religion or sex, and regards merit as the only acceptable criterion for determining student admissions and staff appointments.

The Act under which the Natal University College was established in 1910 imposed no colour-bar or racial restrictions on student admissions, and in 1936 classes for "non-European" students were in-augurated in Durban, while a Medical School accepting African, Indian and Coloured students started classes in the same centre in 1951.

A government ruling that the Medical School should not be open to "white" undergraduates was the first major inroad upon the University's freedom to determine its own admission criteria. In 1957 this freedom was further threatened when a Bill was introduced in Parliament which provided for the removal of the Medical School from the University's control, and for the exclusion of "non-white" students from the University's other Faculties.

This Bill was condemned by the Council and Senate of the University as "a flagrant breach of the principle of university autonomy and as a grave assault upon academic freedom", and the academic staff of the Faculty of Medicine resolved to resign en masse if the Bill was enacted.

As a result of these and other protests, the government withdrew the proposal to remove the Medical School from the University's control; but in 1959, despite continuing opposition, the Extension of University Education Act was passed, in terms of which ministerial permission was required for a student who was not classified "white" to register in a Faculty other than the Medical School.

By the time of this enactment, the University of Natal had developed into the largest university centre for "black" (African, Indian and Coloured) students south of the equator.

As an expression of its resolve to recover the right to admit students on merit, the University inaugurated, both in Durban and in Pietermaritzburg, an annual Academic Freedom lecture, at which the following affirmation is made.

> We are gathered here today to affirm in the name of the University of Natal that it is our duty to uphold the principle that a university is a place where men and women without regard to race and colour are welcome to join in the acquisition and advancement of knowledge and to continue faithfully to defend this ideal against all who have sought by legislative enactment to curtail the autonomy of the University.
>
> Now therefore we dedicate ourselves to the maintenance of this ideal and to the restoration of the autonomy of our University.

In 1983 further legislation was passed relating to university admissions. The effect of this was to replace the 1959 "permit" system by a system, under which the Minister of Education may determine racial "quotas" in respect of student admissions to the University.

So far no quotas have been imposed, but the Medical School remains closed to white undergraduates in terms of government policy.

The University remains wholly opposed to racial inequalities in education, and looks forward to the day when it will be free to serve the cause of learning as a truly open institution.

THE UNIVERSITY OF NATAL COMPUTER SERVICES DIVISION

Up until the end of 1987 the Computer Services Division was split along support and functional lines. For instance, there were different groups of people responsible for Academic and Non-Academic computing. Early in 1988 the organisational structure of the CSD was changed to do away with what were now considered artificial distinctions between Academic and Non-Academic computing services. With the proliferation of micro computing, the expansion of the community network and the mix of applications being run, it no longer made sense to support our various customers with different groups of people. The support for word processing, for instance, is the same for Academics, Non-Academics and students.

The next step was to create an understanding in all CSD staff of our 'service' orientation and the need for 'customer focus'. A 'help desk' was put into place to

respond to our customers problems, enquiries, etc. (Currently we have +-1200 calls logged per month). Our customers are approximately 3500 staff (academic and non-academic) and approximately 13000 students. A user consultancy department was established mainly to support all aspects of the micro computing development spontaneously taking place all over the various campuses.

At this time (early 1988) we ran various mainframes, mini-computers and numerous micro computers. These had been installed over a number of years without a proper strategy plan being compiled and/or followed. Some of the hardware equipment had been 'donated' free of charge but the cost of ownership was now so high as to make the continued running of these 'freebies' uneconomical.

Many of the Administrative Application systems were very old (circa 1971), had been developed in-house and were still mostly batch orientated. The cost of maintenance of such systems was, of course, very high.

Very recently, and in accordance with many American Universities' policy, the responsibilities for the telephone systems and the printing shop were included in the Computer Services Division portfolio. The reasoning behind this is the computer-based integration taking place between all methods of communication. (Voice, Data, Text, Graphics).

A NEW PHILOSOPHY AND STRATEGY

Whilst not exactly in a mess, the state of affairs regarding computing at the University was not ideal early in 1988. Added to other problems was the fact that the Government subsidy to the University was being cut each year and there was not a lot of money to purchase new computer hardware. Much of the money allocated to the CSD budget was spent on just keeping things going.

How to break this log jam was therefore the problem. How do you replace two large mainframes, old systems and improve the network without incurring huge capital expenditure? There is a riddle often quoted in S.A. that asks, "How do you eat an Elephant?" The answer is, "Bit by bit and slowly".

So here we were facing the following problems:

1) High costs associated with running two mainframes and various different mini computers.

2) Extensive mix of skills needed to support the various operating systems and development languages.

3) Users (especially the non-academic users) not satisfied with the current system.

4) Largely batch orientated administrative systems.

5) The expressed desire of the majority of the larger user departments to be more involved in running their own systems and be responsible for their own data. The bureau type service available was not satisfactory to them.

6) The difficulty in linking (integrating) the various systems, running on the various computers, where such systems had been developed in isolation.

7) Backlog of applications development.

Coupled with these no doubt familiar problems was the great explosion in the acquisition of micro computers with the tendency of users to 'do their own thing'.

To overcome all of these perceived problems it was decided to adopt a philosophy of Departmental Computing.

This meant we would identify areas of activity that could and wished to have their own computer facilities physically located in their department and "put the power where the action is".

We believe there is a difference between departmental and distributed computing. Simply, our perception of distributed computing is that you may have many branches of an organisation all running the same application systems on individual computers, maybe linked together and/or to a central head office computer. Typically this could be a purchasing, order entry and inventory control type application.

Conversely, in departmental computing we see each department running its own unique application such as Financial Accounting, Personnel, Salaries, Library Management and Student Administration. However the need to link these individual computers together and integrate these unique applications and data still exists and must be possible. The 'corporate database' although not physically housed in one location must appear to be so to the users.

Which departments get their own computer and which departments share computers is determined by functionality, efficiency, effectiveness, ability and cost benefits.

One of the major advantages of such a philosophy is that each departmental computer becomes application orientated, serving only that departments' objectives. There is no competition for inter departmental resources and it spreads risk in terms of hardware failure (one computer being down does not effect other departments). The total responsibility for the integrity of the data stored on a computer rests with the department. They become the owners of their system and the data.

Having determined a computing philosophy the next step was to develop a strategy which is being implemented in the following manner:

1) Install mini computers (and/or micro based local area networked clusters) on a departmental basis linked by a local communication network, linked in turn to a wider national and international network.

2) Only support non-proprietary operating systems e.g. PICK, UNIX and MS-DOS.

3) Work on a partnership basis with a limited number of preferred vendors.

4) Install on-line application packages whenever possible and limit in-house development to only really unique applications.

5) All application packages and any in-house developments to be based upon, if possible, a common database system.

6) All requests for computer hardware and software to be based upon a positive cost benefit exercise.

7) Replace the two mainframes by the introduction of departmental computing as soon as possible.

8) Teaching of students to be done by use of micro LAN laboratories and lecture facilities shared between academic departments.

9) Departmental academic research and specialized teaching, not covered by the above, to be achieved using any or all of the following; departmental micro computers; departmental mini computers; mini or micro based LAN's or; via the communication network to other organisations where more power and/or specialized software exists. (Buy external resources).

10) Implement a standard office automation system throughout the University.

These benefits are clearly arguable but in our view, suit our situation of: Placing the power where the action is; the users are more involved and responsible for their own computing; it is cost effective, it spreads the risk in terms of down-time and other disasters; it accommodates incremental spending as we swop from the old to the new; there is standardization of support resources; the use of packages will reduce the backlog of applications development, and it is a flexible approach that can respond to the constant changes that the University, technology and user requirements are undergoing.

In our view these are real and tangible benefits of departmental computing.

However one of the dangers inherent in departmental computing is the tendency for each department to set up its own data processing staff, with end users computing turning into end user development.

One of the services that is not being decentralized is the Computer Services Division. Our role is changing from data processing to one of consultancy, facilities management and training, making users self-sufficient but acting in a co-ordinating role in terms of hardware and software acquisitions, supporting users in terms of operating and application system software and managing the facilities where a department does not have the resources or simply wishes not to get involved in any other aspect of computing. The control and maintenance of these services will also fall into our portfolio.

Because of the strategy adopted we will be able to reduce substantially the current need for a wide spread of specialists to support the numerous operating systems and operation software, concentrating our skills into much fewer areas of speciality.

PRACTICAL IMPLEMENTATION

Having presented the Philosophy and Strategies to the various computer-related steering committees and sub-committees and getting approval from the Senate and Council of the University, we were now ready to put it all into practice.

We are perhaps fortunate that in S.A. there is a company with a specific vertical market, that of Tertiary Software Applications Systems covering most of the administrative needs of a University. This is a 'packaged system consisting of modules covering the following areas:

Financial
Student Information
Personnel
Fixed Assets
Space Inventory
Receipting and ad-hoc Payments
Management Information
Capital Project Management
Residence Administration
Bursaries and Loans Administration

All of these modules are integrated and form the composite data base from which all the necessary returns that need to be supplied to earn a government subsidy can be extracted with ease. (An enormous task without proper information systems in place).

This system currently runs under the UNIX operating system and the ORACLE relational data base.

If the Computer Services Division had followed the "traditional" route of producing a statement of user requirements, going out to tender, evaluating the tenders and then finally selecting a software/hardware solution it would have taken quite a few months and cost quite a considerable amount of money.

Since the system as described above seemed to offer a viable solution and was completely in line with our philosophy and strategy we adopted a unconventional approach. We asked the various Administration sections to examine the 'package' and then tell us if it was suitable for their needs and if not why not.

Within a month we got the go-ahead to purchase the software and start implementing the departmental computing strategy.

A master plan was then drawn up and implementation commenced.

The distribution of the Administration Computers as currently planned is;

Pietermaritzburg Library	- Already installed (PICK)
Durban Library	- Already installed (PICK)
Financial Departments	- Already installed (UNIX)
Personnel Department	- Already installed (UNIX)
Student Administration Dept	- Due in July 1990 (UNIX)

The Library computers are based upon the PICK operating system since the library software package we adopted is running only in this environment.

All of these computers are linked via our communications network.

If all continues to go well by the end of 1990 the following main objectives will have been met:

Mainframes closed down.
Support of only PICK, UNIX, MS-DOS. Single relational data base system for all administrative applications.
Libraries on all campuses running on the same system, PICK based, that can integrate with all other systems.
Comprehensive communications network in place.
Modern, on-line systems in place.
A distributed corporate data base in place from which management (both academic and non-academic) can obtain executive information.
Satisfied users (if that is ever possible).

COMMUNICATIONS NETWORK

The Computer Services Division has a very simple (!) philosophy regarding computer communications.

That anybody, from any terminal or micro computer that is part of a LAN, that in turn has a gateway into the backbone University network will be able to have access to any other computer on the network and any other computer on international networks.

Thus any Dean, for instance, can get access to the financial system, make enquiries into the library data base, get information on a student and/or send messages to a colleague in any part of the world. All from one terminal or micro computer.

Easier said than done, but we are already a good way towards achieving this and by the end of 1990 we will be even closer to our goals.

Via our existing LAN installations (+- 600 micro computers in +- 12 LAN's) gatewayed into our CASE-based backbone network and the +-250 terminals directly connected to the CSD via their host computers we have extensive crosstalk between systems and applications.

At the time of writing a project team is investigating a new communication network strategy integrating Voice, Data, Text and Graphics. This investigation covers such things as replacing our switchboards and the introduction of ISDN for instance.

There is in place a backbone network in South Africa which is a collaborative venture between the Council for Scientific Research and the majority of universities. This is called UNINET and its goals are to provide common access to data links between the computer centres of member organisations so as to facilitate the provision of remote login, file transfer, electronic mail and other network services to the South African scientific and engineering research community as well as the storing of information at an administration level.

SUMMARY

With the departmental route any user from any point in the communications network should, with the right authority, be able to access up to date information covering all aspects of University/corporate endeavour.

We make no claims that this is the only approach and indeed that it would be applicable in any other organisation. However bearing in mind the key factors of organisational culture, resources available (both financial and human) and the need to change our current systems we are convinced that this route gives our users the best value for the very limited money available.

Any organisation that does not have a computing philosophy, defined strategies and a tactical plan that is in line with the objective (and the culture) of the organisation will end up with unsatisfactory, costly and doomed-to-failure systems.

At least after many years of unplanned computer growth we now have philosophies and strategies in place and have defined our objectives.

Since this is a Management Symposium the management aspects of our departmental computing have been emphasized and the technical aspects skipped over. For too long we have been 'hooked' on the technology whereas we should be concentrating on the business management of computerization.

Software considerations are now more important than hardware and the cost of ownership far more important than the cost of acquisition.

Users can no longer be fooled by the 'mystique' that the computer industry has carefully nurtured over the years, and they now want to be part of the action. Departmental computers is one strategy to follow that ensures that this happens.

Great political and cultural changes are taking place in South Africa and we believe that the University of Natal in both attitude and deed is preparing and will be ready (indeed is influencing) these changes. The Computer Services Division, as a service department, is looking forward to the future.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

Presented by: William Pugsley

Job Title: Chief Executive Officer

Organization: Perwill Group of Companies

Addresses:

Perwill Inc.,
5053 La Mart Drive Suite 101
Riverside
CA 92507
U.S.A.

Tel: (714) 683 7920
Fax: (714) 787 8789

Perwill Group
Underwood
Swaines Hill
Odiham Road, Alton
Hants. GU34 4DP
United Kingdom

Tel: +44 256 862003
Fax: +44 256 862168

Perwill Networked Solutions Pte Ltd
Innovation Centre #02-201
NTI Nanyang Avenue
Singapore 2263
Tel: +65 261-1620
Fax: +65 261-9410

ABSTRACT

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI) is a presentation designed to provide a comprehensive insight into what Electronic Data Interchange (EDI) is.

It examines typical Business Relationships and looks at the historic and International development of EDI, with a brief review of what Standards are available today and those that are emerging Worldwide.

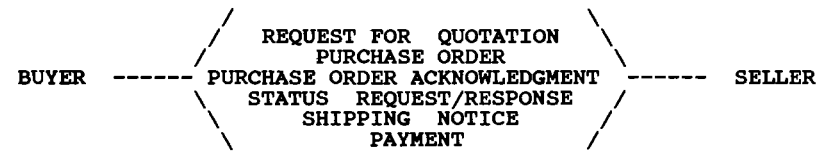
The paper then looks at Companies that are actually using EDI and develops the reasons for using EDI for Business. This is followed by a review of the steps to be taken towards EDI Implementation.

The main presentation then concludes by looking at what the author believes are the many benefits of EDI, closing with a series of questions that would be practitioners need to consider before embarking upon EDI.

The formal paper is then followed by a detailed class exercise that concentrates on the ANSI.X12 EDI Syntax, which depending upon time, will allow the attendee have an understanding of some of the detailed issues associated with data mapping. Work sheets for the class exercise will be provided during the tutorial.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

The typical business relationship between buyers and sellers (which will be referred to more correctly in Electronic Data Interchange (EDI) Terms as business partner) involves a set of transactions that could typically look like:



To begin to realize the benefits of EDI, it is important to consider the current Business Practices for moving the above data between partners. Typically, a request for a quotation may be printed on the buyers machine, vetted by someone and put into an envelope and consigned to the local postal service. This service then delivers the envelope to the seller, who opens the envelope, checks the information and may then key the data into his machine. This can result in another printed piece of paper that is checked by someone, put into an envelope and passed back through the postal system to the buyer. The buyer then checks the contents of the quotation, perhaps keys it into his purchasing system and prints out a purchase order which is validated, put into an envelope and passed to the postal system for conveying to the seller.

In the early stages of improving business practices telex systems were used to speed the timing of delivery of information. However, this did not remove the need of keying-in the data. Did you realism that:

70% PLUS OF COMPUTER KEYED INPUT (ON AVERAGE) IS TAKEN FROM A COMPUTER PRODUCED DOCUMENT

What is really wanted is the ability to have your machine "talk" directly to your business partners' machine. Not only does this remove the problems of time associated with traditional methods of delivering data from one company to another (consider the postal system), it also can remove the need for the re-keying of data.

So Electronic Data Interchange can be described as:

THE COMPLETE COMPUTER APPLICATION TO COMPUTER APPLICATION EXCHANGE OF INTERCOMPANY (OR INTRACOMPANY) DOCUMENTS AND INFORMATION

However, a number of factors need to be considered before a business can simply implement the above scenario. Unfortunately,

THE REAL WORLD IS NOT THAT SIMPLE

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

The first consideration is what kind of communication/ /connectivity protocol can be used. Where your business partner has the same hardware as yourself, it is practical to consider that there will be no problem with the connectivity issue. In the case of Hewlett-Packard Series 3000's, you can use DSN, WAN, NS or even RJE (at a price of course). Yet your partner may not have the same machine as you. They could be using IBM Mainframes, UNISYS, NCR, DEC or ICL so what do you do then?

Assuming that you overcome the connectivity problems, you then have to consider the timings of the connection. The fact that your purchasing system is ready with its Purchase Orders at 4.00am does not mean to say that your supplier can actually receive them at that time, let alone process them. You may now have to plan changing your schedules to accommodate your mutual needs.

This problem of connectivity can now be compounded by the numerous types of participants. It is relatively easy to agree formats with business partners in the same or similar industry to your own. What happens when you find that you start dealing across industries. Historically different industry sectors have been able to agree formats of information for exchanging between them. You now have to think about what an Insurance Broker, a Rail Carrier or indeed what your Supplier would like (or indeed want) to see in the information passing between you and them.

One way of removing most, if not all, of the above obstacles is by using

VALUE ADDED NETWORKS (VANS)

VANS service often allows for the different types of connectivity needed by different businesses, whether they use HP, DEC, IBM etc. As they normally provide a store and forward facility, the question of timings is immediately removed but perhaps more importantly they can provide a

PUSH and PULL

facility. This allows you to decide when you want to connect, whether you are passing data into their network or inquiring on what data is waiting for you. If you then want to receive that data you execute a command set to allow it to be sent into your machine. With direct connect to a business partner machine one of you has to allow read access to the other - an area potentially dangerous (but you need to talk to your auditors about that).

However, having overcome the connectivity, timing and security issues you now have to consider the next stage - the documents themselves. Each document type has its own format. This problem can be overcome by accepting that each document does indeed have its own format and identifying an environment that allows for the different formats that are available, ie. Purchase Order, Invoice, Delivery Advice etc;

The next problem is that no two invoices are identical! Every Company that I have worked with has a different format for its own invoices, this includes separate divisions within a single Company! The information may be similar but you will rarely find the ship-to address in the same location on the different forms. But it is true to say that most of the information is consistent in that an Invoice will contain a Bill-to Address, a Remit-to Address, perhaps an Order-by Address, Registered Office Address, details of items supplied (to include product identifier, quantity, description, unit price, extended price), discounts, terms etc.

Having established the mechanism for transmission of data within an EDI environment, ie. machine to machine, you still have to overcome the problem of format of information. To allow the processing of transported data,

EDI IMPLIES THAT A STANDARD IS USED TO PRECISELY DEFINE ELEMENTS OF THE ELECTRONIC MESSAGE

Again, the real world is not that simple in that there is not just one standard. Some of the currently used standards for the formatting of messages include:

ANSI.X12	**TRADACOMMS**
TDCC - UCS	**ODETTE**
WINS	**EDIFACT**
CALS	**SPEC2000**
etc	**etc**

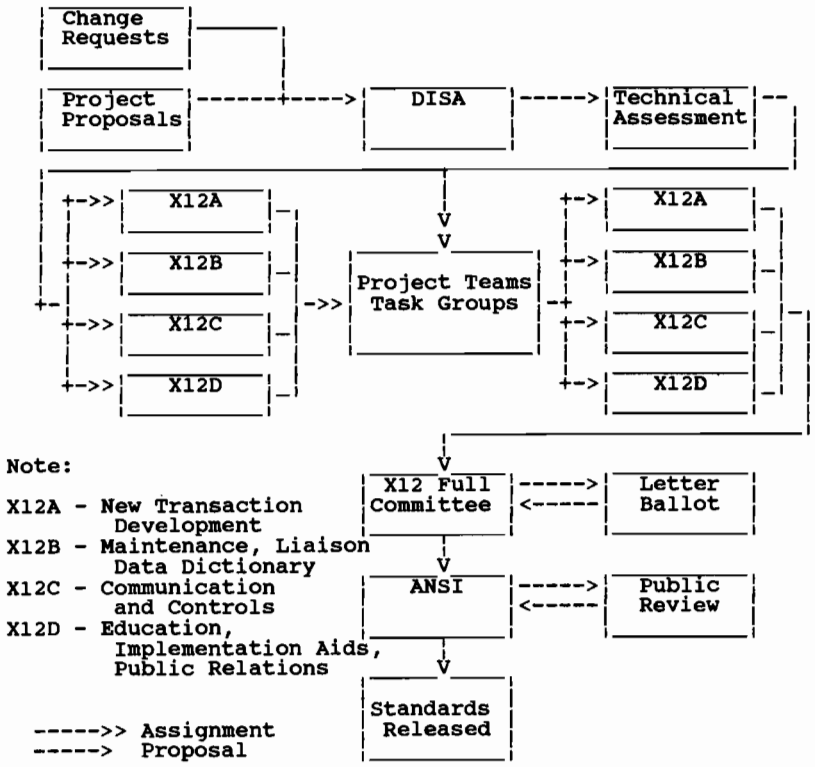
It may be appropriate to consider the historical development of EDI. In the USA, several standards evolved to meet the needs of the different Industry Sector Users. The United States Grocery Industry adopted UCS, the Warehousing and forwarding groups adopted WINS and the Transportation Group employed TDCC (Transportation Data Coordination Committee).

In the late 70's it was recognized that it was no longer appropriate to have the different groups in the USA and they became consolidated into what has been identified as ANSI ASC X12 (American National Standards Institute Accredited Standards Committee X12 - where X12 is a simple project team designation).

ANSI X12 Development took the following Route:

1978	Founded by CRF
1979	Chartered by ANSI
1981	Initial Standards Completed
1983	Five Standards Approved
1984	Alignment of Data Dictionaries with JEDI
	* 5 Standards Updated
	* 13 New Standards Completed
1986	Approval of New and Updated Standards

The ANSI development process continues with at least three meetings a year. The council for ANSI Development has created the following structure for all development:



TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

In the early 80's in Europe, the body that became known as ODETTE (Organization for Data Exchange by Tele Transmission in Europe) also began work on defining the needs for Business Communication Messages. Although originally set up for the Automotive Industry it is used for a number of industry groups outside that sphere.

In the meantime, in the United Kingdom, the ANA (Article Numbering Association) in conjunction with a number of Industry Groups embarked upon the development of TRADACOMM (Trading Data Communication). This is now widely used in the United Kingdom by many types of Businesses.

The most important development to-date has been the creation of EDIFACT (Electronic Data Interchange for Administration, Commerce and Transportation - or Trade). This single Standard has emerged as being the only true multi-industry / multi-national Standard that is being embraced throughout the World.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

To consider the rapid development of EDIFACT it may be appropriate to consider its History.

1985 UN/ECE GTDI & ANSI X12

1985 Sept. UN/ECE encouraged to merge GTDI with ANSI ASC X12

1985 Nov. Experts meet from CANADA, Europe and USA

1986 March North Americans and Europeans meet to discuss emerging Standards

1987 March UN appoints EDIFACT Rapporteurs & approves Syntax

1987 April Preliminary Message Guidelines Developed

1987 June Draft Message Design Guidelines Available

1987 Sept. ISO Approved EDIFACT Syntax

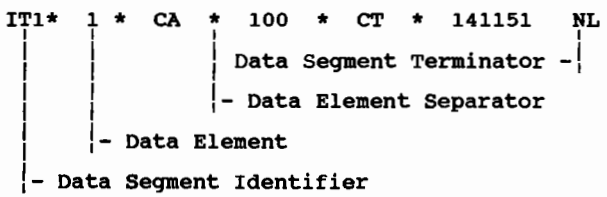
Having dealt with the development of Standards it is now appropriate to consider precisely what the Standard Data looks like. For the purposes of illustration, the ANSI X12 Standard will be used (as it is most like the EDIFACT Standard) when looking at EDI Terminology:

The EDI transmission can consist of several FUNCTIONAL GROUPS which may be likened to sets of Invoices, Purchase Orders etc (Different Documents). Within the FUNCTIONAL GROUP will be the individual TRANSACTION SETS (the Invoices, as distinct from several document types). Yet each TRANSACTION is made up of DATA SEGMENTS, which can be repeated (or LOOP), for example Invoice Address, Invoice Item Line etc; and finally DATA ELEMENTS that are the unique fields within the DATA SEGMENTS. Here a DATA ELEMENT could be a Quantity, Unit of Measure, Description, Unit Price etc; It may be useful to consider DATA ELEMENTS as fields within Records (DATA SEGMENTS) that are held within a file of Invoices (TRANSACTION SETS).

From here it is possible to build a DATA SEGMENT which may look like:

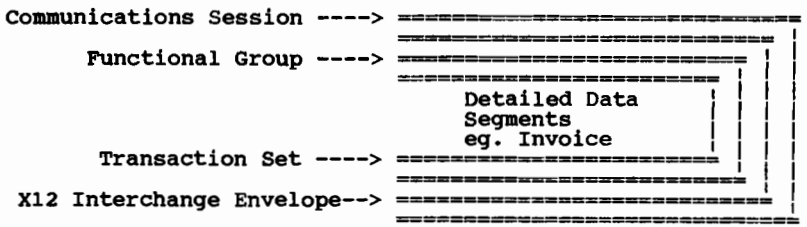
	Quantity	Unit of Measure	Unit Price	Catalogue Number	Description	
IT1*	Quantity Invoiced	Unit of Measure	Unit Price	Product ID Qualifier	Product ID	NL
	1	Case	$1.00	Catalogue Number	141151	
IT1*	1	CA	100	CT	141151	NL

STRUCTURE OF AN X12 DATA SEGMENT



Having described the actual content of an EDI Message we have to now consider how the data gets from one location to another. EDI messages are delivered using an Electronic Envelope.

The following illustration presents the above information within the constraints of the X12 INTERCHANGE CONTROL.



TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

WHO USES EDI NOW

The United States Grocery Industry estimated that it has saved over US$150,000,000.00 during 1987 by using EDI. These savings have been realized in both improved communications leading to better availability of Stock (which normally has a short shelf life) and reduction in errors of re-keying data.

ICI have been using EDI in various forms since 1982 and have been able to improve profitability as a result of the savings made in not spending time on errors.

Courtaulds (Clothing Manufacturer) and Marks & Spencer (Major Retail Outlet) at one time used Couriers from Stores to the Manufacturer that cost upwards of US$450,000.00 per year. The Couriers were used to ensure the timely notification of demand for a replenishment product. This ensured that orders could be processed overnight with delivery often the next day. Using EDI they can advise on requirement and often receive replenishment the same day. Apart from improved Customer Service (by the Manufacturer), Marks & Spencer no longer pay for the Courier service!

WHY MOVE INTO EDI

Perhaps the usual reason is savings in money! The examples given earlier go some way to explaining cost benefits but the most significant that we have encountered is one organization that issued several million purchase orders a year to its suppliers. After analysis, costed each purchase order at about US$50.00 each (most companies can cost about US$30.00 per order). On further analysis it was discovered that 1.3 million purchase orders were actually being sent to seven suppliers. This represented a cost of US$65 million per year on purchase orders to those suppliers. After one year, having implemented EDI with those seven suppliers the exercise of costing the purchase orders was done again and it was found that they now (for the 1.3 million) cost on average US$15.00 - giving a saving of US$45.5 million!

Better Business also comes to mind when considering the benefits of EDI. Levi Strauss USA, clothing manufacturers, were waiting up to twelve weeks for feed-back to its corporate offices on sales. If a product was selling faster than planned, it meant that production was not being geared up to meet the demand in time. By the same token, if a product had "bombed", the knowledge that it was not selling, was not being received until too late and as much as twelve weeks of volume production had been made. By integrating EPOS (Electronic Point of Sale) data direct to EDI messaging to corporate offices, Levi Strauss now know within two weeks how a product is doing!

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

More efficient Business was the objective of Littlewoods Catalogue Group (a company that sells via agents direct into homes). Littlewoods wanted to improve its service to customers by making sure that a customer did not have to wait more than two weeks from placing an order to delivery. Initially, certain suppliers were targeted and a plan was put in place whereby the suppliers wrapped and delivered product' direct to the end client (as if direct) upon notification of demand by Littlewoods. When dispatched, the supplier advised Littlewoods who then billed the agent and paid the supplier. From the original objective of cutting down lead time for delivery evolved a far greater reliance on direct delivery by suppliers. This reduced the amount of inventory moving into Littlewoods warehouses, cut back on the need for warehousing (with the resultant sale of unused buildings) and did not tie capital up in inventory. Until it was moved from the supplier warehouse - it was not paid for.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

GETTING STARTED

Before considering how to actually start the implementation of EDI, it is important to examine the reasons for your Business involvement in what can only be described as a changing Business platform.

There are essentially two ways that companies actually get involved in EDI, and they are:

REACTIVE
(The Wrong Way)

and

PROACTIVE
(The Best Way)

In the first instance, a Business Partner or Government Agency dictates both the methods and timescales that you have to adopt. Although it is possible to react quickly mistakes can be made that may take a significant effort and time to correct at a later date.

To illustrate this, one company comes to mind that called a supplier of EDI Software on 3rd January 1989, met with their consultants on 4th January, gave an order for software on 6th January, commenced installation of software on 16th January and received "live" Purchase Orders, via a Value Added Network (VAN) on 17th January 1989, straight into their Sales Order Application! But they lost out on negotiating with their Customer what they both wanted to do and what they were able to do. They happened to be lucky as if they had not been able to meet the timescale of their client they would have had to wait until April with the implied threat that they would receive less orders in the meantime.

With PROACTIVE planning, you can decide most of the steps that I will outline in the main presentation. You go with a position of strength to your Business Partners and will be able to present your needs to them. These needs could include dealing only with your Product Codes, rather than theirs (consider the implications on your existing systems to change from, say a five digit product code to their thirteen digit code).

WHO GETS THE MOST BENEFIT?

Hopefully, having decided to operate Proactively, you now need to look internally within your own organization to see who is going to derive the greatest benefit.

Depending upon who has been "SOLD" on the idea, will depend where you have to invest the most effort in internal education.

In Large Multi-national Companies, the Financial Controllers perceive the benefits in receiving Intra-company spread sheet data within hours of the end of financial accounting periods so that they can be consolidated into a single corporate statement. In other organizations, Financial Controllers will be looking at speedier payments by customers as a justification for implementation.

Purchasing Managers will be looking to reduce the unit costs of preparing and transmitting Purchase Orders to suppliers, with the associated benefits of reduced errors (because there is no longer the need to chase errors caused by re-keying of data), electronic messages can be acknowledged so reduced questions of "did you get it?", through to being able to monitor actual product availability (EDI catalogues with price and availability).

Manufacturing Controllers will see the benefits of being able to reduce stock holding, allowing EDI to augment their Just-in-Time process controls (the raw materials arrive just before they are needed) through to being able to quickly place demand on suppliers when unexpected production increases warrant it.

Sales Managers are looking to be more competitive and provide more cost effective product and service. EDI allows them to receive orders directly from Customers and respond to the customers needs in a timely manner.

Planning Managers can organize forward production in conjunction with raw material suppliers and end customers to optimize the throughput of the organization. No longer does significant "padding" for lead time, information processing and errors have to be allowed. EDI, when integrated in all Business functions, can literally assume that goods will be delivered one hour before they are needed (but I would recommend that you allow at least two working days!), compared to the many weeks normally allowed.

SET UP THE PROJECT TEAM

The most successful implementors of EDI that I have encountered are those companies that allocate a full time EDI Program Manager. This person ensures that they become educated in as many facets of EDI as are necessary for them to ensure that the best interests of their company are met.

The EDI Program Manager must be allocated reasonable budget, but be answerable to senior management (often Board Level) for the progress and planning of the EDI implementation.

User Managers and Staff of those Business areas that are decided to be involved in EDI should be co-opted onto a planning/implementation committee. They will work under the auspices of the EDI Program Manager to the various tasks that are set for them. These tasks may include setting up internal training.

The final element that is necessary for the support of the implementation is the backing of the Board, or at the very least the senior executive management. Without this backing it may not be possible to push through the fundamental changes that can occur with EDI. How often have you heard the expression "But I need this bit of paper to do my job!".

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

PROJECT PLAN

Having decided that you have (need?) to be involved in EDI, you have to determine who you are going to work with. This is often called:

1. Targeting Trading Partners

- **BUSINESS SENSE**

One company that we have encountered is an organization that issues several million purchase orders a year to its suppliers. After analysis, costed each purchase order at about US$50.00 each (most companies can cost about US$30.00 per order). On further analysis it was discovered that 1.3 million purchase orders were actually being sent to seven suppliers. This represented a cost of US$65 million per year on purchase orders to those suppliers. After one year, having implemented EDI with those seven suppliers the exercise of costing the purchase orders was done again and it was found that they now (for the 1.3 million) cost on average US$15.00 - giving a saving of US$45.5 million!

- **PRESSURE**

If you are in the reactive mode of operation (for an example a supplier to the North American Motor Industry) you may find that your Partners have already been selected for you. These situations have increased radically in the United Kingdom where organizations like British Coal, The National Health Service, British Sugar etc; have told their suppliers to trade using EDI. As indicated earlier, in this mode you will normally find that the Partner has a sophisticated knowledge of EDI and they will invariably impress upon you their requirements, with limited regard to your resources or needs.

In a few situations, you may have received a questionnaire or a general inquiry from Business Partners, who like yourselves are just beginning to look into EDI as a means of improving their own Business.

- **THE QUESTIONNAIRE**

Given the situation where there are many potential Partners, for example, you can identify thirty who may represent as much as fifty plus percent of the paper flow you have in a given sector, you need to be able to isolate those that you could attempt to work with. To do this you can consider sending out a questionnaire (of which a sample follows). Note the content of the last question.

PROJECT PLAN

SAMPLE EDI QUESTIONNAIRE

1. Have you heard of EDI? Yes No

2. Do you know what Standards there are in EDI? Yes No

3. Are you using or plan to use EDI? Yes No

4. Would you be interested in investigating EDI with us? Yes No

5. What Documents would you be interested in Trading with us using EDI?

   Purchase Orders ☐ Invoices ☐ Bill of Lading ☐ etc...

6. How many companies are you trading in EDI with now?

   None ☐ 1-5 ☐ 6-10 ☐ more than 10 ☐

7. What Standard / Version / Mark are you using?

   _____ / _____ / ____

COMMENTS_____

_____

_____

_____

Thank you for the time you spent on this questionnaire.

2. Timescale

- QUESTIONNAIRE RESPONSES

Given that you decide to send out the questionnaire, you may find that responses are as low as twenty percent (wait no more than two weeks for responses). You will then have to evaluate them, looking for those with the most complete answers. You may be lucky to have as many as four or five completely answered then you need to look at the answer to number seven. If the "mark" field is blank, perhaps with a comment asking what it is, you have found someone who has a good understanding of EDI (there is no such thing as a "mark" - there is a "variant"). It is those with the more complete answers that you want to concentrate on first.

- EDUCATION PLANNING

There are several distinct areas that need to be considered, each with a varying degree of preparation, but often allowing the "borrowing" of elements from one area to another.

- General Awareness of EDI Seminar for internal staff

- Planned external Awareness of EDI Seminar for Business Partners who you are targeting

- Detailed presentations for Users to identify the implemented changes and who will be affected (if any one is) by those changes

- Data Processing Departments will have to become familiar with whatever mechanisms are put in place to facilitate the usage of EDI

Preparation for each of these can be as little as one day or as long as several months, depending upon the content that it is decided to have (many Trade Organizations and User Communities have prepared presentations that can be borrowed or rented, some vendors will provide assistance with presentations as part of sales support and/or packages).

- UNDERSTANDING

Either you invest the time (and money) in allowing your own EDI Program Manager to become sufficiently familiar with the mechanisms of EDI or you will have to rely on external consultants. Value Added Networks often provide a service of education but they do not invite you to ask questions that they would find uncomfortable to answer!

PROJECT PLAN

2. Timescale continued

- **CHECKPOINTS**

Identify early on what all your targets and dates are. Even if you are unable to fully quantify them, at least ensure that you know what they are. They can include all the items that are part of this presentation, augmented by your own specific needs. Again, Trade Organizations and User Groups are able to assist, often at no or little cost, providing you with all the elements required for consideration.

- **RESOURCE ALLOCATION**

It has already been noted that those companies who succeed in implementing EDI allocate a full time Manager to oversee the activities and ensure that all participants (the co-opted Users and any vendors) are working towards the goal of implementing EDI. The manager is also responsible for initiating and monitoring the Project Plan.

3. Partnership

- **WHO FIRST?**

This question should have already been answered by the activities of step 1. (Targeting Trading Partners). However, always be prepared to consider changing those Partners that you identify early on. You may discover others who have more experience or will create a measurable saving in doing Business with them electronically.

- **WHICH DOCUMENTS?**

I have worked with some companies that have actually looked for an initial EDI relationship where the volumes would be low. The intent being to ensure that the relationship and activity could be closely monitored. However, once the processes were verified, then they sought situations of volume bringing reduced cost.

An example that comes to mind is the situation where one company was sending on average 40,000 Invoices a month to one Customer. It was taking anything up to 120 days to for payment. The Customer was also having to use a data entry group of eleven people to key the invoices in, upon receipt - with its associated delays and errors. EDI seemed an ideal way of improving processing for both parties.

Each month, the full detail of the 40,000 invoices is passed by EDI to the Customer with four batch totals (about 10,000 Invoices per Batch). Four checks for payment are raised within three/four working days plus/minus any adjustments from the previous month. The Customer modified its existing Payables System to receive the detailed invoices directly into it and automatically match against goods receipts. Most of the eleven staff were reallocated to other tasks within the Company, with those remaining concentrating on the mis-matches reported from the Payables System. The earlier payment was justified by the reduced cost of personnel resourcing, the creation of the EDI packages by the Seller, had its cost recovered by the improved cash-flow.

Continued/.....

3. Partnership continued

- WHICH DOCUMENTS? continued

A second area is in time critical activities. It may be imperative that product demand reach a supplier as early as possible such that replenishment can follow quickly. If these situations can be identified, regardless of volumes, then they can be exploited.

Courtaulds (Clothing Manufacturer) and Marks & Spencer (Major Retail Outlet) at one time used Couriers from Stores to the Manufacturer that cost upwards of US$450,000.00 per year. The Couriers were used to ensure the timely notification of demand for a replenishment product. This ensured that orders could be processed overnight with delivery often the next day. Using EDI they can advise on requirement and often receive replenishment the same day. Apart from improved Customer Service (by the Manufacturer), Marks & Spencer no longer pay for the Courier service!

Both examples just presented give improved productivity. In the United Kingdom most "Do-it-Yourself" suppliers now insist that they have EDI facilities with their suppliers. Most of their Business is conducted Friday, Saturday and Sunday and they require re-stocking during the following Monday through Thursday. Prior to introducing EDI they needed to have at least two weeks worth of product on-hand as it took six working days to process the orders using paper.

- GAINING COMMITMENT

I am assuming that you have already obtained your internal company commitment to EDI. The area that you have to address is that of your Business Partners.

They will need to be given good reason for embarking upon EDI. Ford told their suppliers that unless they worked with EDI they would not be their suppliers. One way of gaining commitment! The earlier example of mutual benefit (the 40,000 invoices) also presents a good reason for embarking upon EDI. You will have to look for the good arguments that will sway your Partner into devoting as much time and energy as you do. If all else fails get your Chief Executive to convince their Chief Executive in their next social engagement (over a game of golf?).

Continued/.....

3. Partnership continued

- EDUCATING PARTNER

Who Pays? is a question often asked. Larger organizations (such as British Coal) prepared a complete series of Seminars for their major suppliers at their own cost. You have to evaluate whether the cost of your preparing the education is justified by the potential returns.

4. Existing Application System Changes

- POTENTIALLY LARGEST COST FACTOR

Most Computer Systems developed in the last ten years have been orientated to Online/Interactive processing where Users key information directly into a Visual Display Unit and have the data validated and checked for consistency as they enter it.

Very few systems created allow the entry of Batch Input. Full implementation of EDI for incoming data relies on there being some means of loading information directly into the existing applications. If there is no present Batch Input facility, it will have to be created.

When establishing a Trading Partner to whom you are going to send data, you have to isolate the output data within your existing application. This could typically mean adding an indicator to a Master File (to show that for this Company you send paper, for this second electronic and for this third type, both paper and electronic). You then need to change the print programs not to print the pieces of paper for those companies that now trade with you electronically.

Are the resources available to make any changes needed to your existing systems that could be affected by EDI? You may find that your Data Processing Resources are tied up on other projects. If so, what timescale of availability can they provide you with. You may have to consider (when being pressured) to implement a half way house solution by obtaining an inexpensive PC package that will print out documents that you receive using EDI (for some one to then enter into your system as they do today) and to actually re-enter the information you want to send. However, there are available a number of "enabling products" that may reduce the need for changing your existing systems.

Continued/....

4. Existing Application System Changes Continued

- **NEW SYSTEMS?**

Do not forget new systems that you are planning to implement. Ensure that they are going to be built with hooks in place for your EDI needs. Unless their development is so far advanced, they should be changed before they are implemented.

- **CHANGING BUSINESS PRACTICE**

Take advantage of EDI to re-think your paper processing. EDI is not just about replacing paper, it is a vehicle that allows you the opportunity of changing the way you do Business. Austin Rover Group in the United Kingdom now do self billing. When they receive goods from a supplier, they check that they were ordered and then simply pay. The supplier does not have to raise an invoice!

5. Standards

You have to understand which standard you are going to use with your Business Partner. Of those that are available, the three most widely used are:

- **EDIFACT** used Worldwide

The most important development to-date in the EDI standards arena has been the creation of EDIFACT (Electronic Data Interchange for Administration, Commerce and Transportation - or Trade). This single Standard has emerged as being the only true multi-industry / multi-national Standard that is being embraced throughout the World.

- **ANSI.X12** used North America and Australia

In the USA, several standards evolved to meet the needs of the different Industry Sector Users. The United States Grocery Industry adopted USC, the Warehousing and forwarding groups adopted WINS and the Transportation Group employed TDCC (Transportation Data Co-ordination Committee).

In the late 70's it was recognized that it was no longer appropriate to have the different groups in the USA and they became consolidated into what has been identified as **ANSI** ASC X12 (American National Standards Institute Accredited Standards Committee X12 - where X12 is a simple project team designation).

- **TRADACOMMS** used in United Kingdom

In the United Kingdom, the ANA (Article Numbering Association) in conjunction with a number of Industry Groups embarked upon the development of TRADACOMM (Trading Data Communication). This is now widely used in the United Kingdom by many types of Businesses.

6. Agreeing Messages

- DOCUMENTS

Once you have identified who you are going to work with you need to identify which documents you are going to replace with electronic messages (and which Standard)

In North America, the first document that was created within ANSI.X12 was the Purchase Order. In EDIFACT, the first document created was the International Invoice. General Electric in the USA introduced Spread Sheets to the ANSI.X12 Standard and in the United Kingdom, Way Bills were introduced for TRADACOMMS. You may actually have a Business need that has not already had a message type designed for it (Manifest say), in which case you may care to attempt to design one within the syntax rules (the vocabulary rules) of the Standard you elect to use - although I would recommend against being a pioneer until you have been working in EDI for some time.

- CONTENT

Hopefully having established those transactions (documents) that you want to convey to/from your Business Partner, you now have to ensure that you agree on the contents. Field Values need to be checked as your understanding of ship-date (the day it leaves your factory) may be the day your Partner actually expects to receive it (and if they are three days journey away it could make them unhappy!). A number of EDI messages provide the provision of a product code. Whose code, yours or theirs?

Where you or your Partner' application is unable to provide elements of data that are needed (another item to be considered when planning to change your application), you need to mutually decide the defaults that will be inserted or assumed in the electronic message.

You may also find that the existing EDI message structure does not provide you with all the potential fields that you currently pass to and from your Business Partner. In this event you have to carefully consider what elements you will want to add, and more importantly, how to add them to the message that you are using.

7. Contracts

- **TERMS & CONDITIONS**

Most paper documents actually carry on their reverse the full terms and conditions that a buyer or seller expects to be adhered to in the event of a dispute over service, payment etc. Most electronic messages designed to replace paper documents do not carry this information, it is no longer sent! It is therefore necessary to separately identify all terms and conditions of business as part of a written contract that is executed before you go live!

- **TECHNICAL SPECIFICATIONS**

Many businesses re-key data received from a Business Partner so that it is available on every printed piece of paper that is passed between them. EDI is about reducing the volume of traffic as well as removing the re-keying of information. Some Businesses are now agreeing to the Sent Once philosophy whereby the originator, normally the buyer, identifies a set of specifications with a unique reference. Thereafter, unless there was a variation to the original notes, it was sufficient to simply use the reference, ie. these products conform to reference XYZ123. But such an agreement needs to be documented to avoid difficulties in understanding later on.

- **AUDIT/CONTROL**

Nearly all EDI Standards provide a Batch Numbering capability. You need to decide how to implement such a capability as you might want to track certain transactions, ie. invoices, to each Business Partner. If you undertake to provide an ascending sequence of Batch Numbers (as distinct from unique invoice numbers), they will be able to verify that they have received them from you and whether any transmissions have been lost. You may decide to simply send Batch Numbers in an ascending sequence regardless of who the information is going to.

- **WRITTEN CONTRACT**

Remembering the joke, "It's your Computer's word against my Computer's word", you should put in place a written contract that protects both parties so that it must define all the elements that have hitherto been done using the existing paper media. The contract should take into account all the additional controls that you intend implementing to allow the ready identification of whether batches of messages have been received by a Business Partner.

8. Communications

The first consideration is what kind of communication/ /connectivity protocol can be used. Where your Business Partner has the same hardware as yourself, it is practical to consider that there will be no problem with the connectivity issue. Yet your Partner may not have the same machine as you. They could be using IBM Mainframes, Hewlett-Packard, UNISYS, NCR, DEC or ICL so what do you do then?

- **DIRECT** (perhaps via Diskette)

In the early stages of EDI implementation, a great deal can be proved by simply exchanging messages on magnetic media such as floppy discs or tape. The Bankers Automated Clearing Service (BACS) in the UK has been successful in using this method for more than twenty years.

- **VALUE ADDED NETWORK** (VAN)

The most common way of providing connectivity between Business Partners is through a VAN that has been established to provide Store and Forward of EDI messages. This allows you to decide when you want to connect, whether you are passing data into their network or inquiring on what data is waiting for you. If you then want to receive that data you execute a command set to allow it to be sent into your machine. With direct connect to a Business Partner machine, one of you has to allow read access to the other - an area potentially dangerous (but you need to talk to your auditors about that). A further advantage of using a VAN, is the provision of an independent audit control of messages sent and received.

- **INTER-CONNECTIVITY**

Assuming that you have selected a VAN to convey your Business data, do they actually cover the Business or Geographic areas that you need. Assuming that the service has been set-up on a Country only scale, do they plan to connect to other VAN's allowing you deal with Business Partners, Holding Companies or Subsidiaries in other Countries?

- **PROTOCOL**

Whether you are connecting directly to a Business Partner or through a VAN, does your machine have the capability of generating the necessary protocol for the connection? In this context, protocol is the term applied to the means by which data can be sent via a telephone line.

9. Cost Options

- PC BASED START-UP

This method is often employed by companies who are interested in getting to know more about EDI without having to make major investments in changing or adding to existing Computer Power. It allows the isolation of connectivity (whether direct or indirect via a VAN) to a PC which is rarely used for application processing. It is also often less expensive to have the VAN connection in the first place (VAN's tend to charge a scale of joining fees associated with the type of machine that you use, not just the volume of documents that you intend to transmit). Unless your volumes warrant the usage of a larger processor, the cost of a complete package, including communications facilities, is often very modest.

An over-riding consideration when choosing a solution is whether the introduction of a PC product will provide upward migration/compatibility to the machine that you may eventually want the software to run on.

- MINI-COMPUTER

If you already have a mini-computer you can implement various means of providing in-house EDI facilities within it. Some major organizations actually buy a mini-computer just to provide an EDI service to their mainframes, again with the advantage of isolating processing from the main application machines. Regardless of the price of any software you use, the potential additional costs to add specific communication facilities to a mini-computer are not modest!

- MAINFRAME COMPUTERS

Software tools are available that will allow the full introduction of EDI on virtually any mainframe. However, the costs of that software are directly in proportion to the cost of the machine - high! Also, adding communication facilities to a mainframe are not cheap.

10. Translation Software

- THIRD PARTY SERVICE (VAN?)

Most Value Added Networks in the United Kingdom will happily provide the service of taking existing application data that you want to pass to a Business Partner and convert it for you into an EDI Package. They are also prepared to do the reverse process for you, ie. take an EDI Package and convert it into a format suitable for direct entry into your applications. As with all things, you pay a fee for each "conversion program" they write and an additional premium for every transmission that you make which involves this process. If your volumes are high, this can become very expensive.

- IN-HOUSE DEVELOPMENT

Given that you have your own Data Processing Department containing Programmers, you could always write translation software yourselves. If you choose this option, be prepared to assign a Programmer to maintain it for the rest of your usage. There is one unchangeable fact about EDI, it is forever changing! Standards continue to evolve and new messages are being created.

- PACKAGED SOLUTIONS

The final alternative is to actually purchase; licence or rent software for your machine that will provide an EDI translation capability. There are a number of products available for most machines that handle the main Standards used in the World today. Some products available are very low cost (and function) whereas others, more expensive offerings, provide very comprehensive solutions. Work with your industry groups or hardware suppliers to identify what is available.

11. Pilot Running

Having decided which Standard to use, the software or facilities and with whom, you now have to actually test out what you have planned.

- TEST DATA

The preparation of Test Data is an essential step for any EDI implementation. The data itself must fully exercise both the facilities that you have set-up and the facilities used by your Business Partner (who is now your Trading Partner). It must contain good as well as bad data to see what happens when bad data is encountered.

- PRODUCTION DATA

Having satisfied yourself that the Test Data can be processed, attempt to push through as much Production Data as possible. You may find that what you thought was produced by your existing systems (and thus catered for in the test data phase) is not in fact produced!

- USER INVOLVEMENT/TRAINING

One of the purposes of EDI is to remove paper. Have you prepared your Users for No More Paper? Some will insist that they are unable to do their job without a certain piece of paper. If you have managed to remove the paper, what really is the job function that is being done? Could it be that you will have to extend your application systems to provide an online look-up that has never been asked for as the paper provided the necessary warning for an action? Explain to the Users at every stage what is being done that could affect the way in which they work. KEEP THEM INFORMED!

- CHECK AUDIT CONTROLS

When you send fifteen invoices to a partner, you need to be assured that you did actually send them and as importantly that they received them. Some systems provide an automatic control mechanism that will not only verify what was sent (and when it was received) but will allow the monitoring of information like Invoice Numbers. Many Countries insist on being able to have access to historic invoicing data. Will you still need to create paper copies to satisfy revenue authorities or Auditors?

Continued/.....

11. Pilot Running continued

- **VERIFY RESULTS**

Do not assume that just because you sent fifteen invoices and your Trading Partner acknowledged receipt of fifteen that it was successful. We once saw a situation where a translator accidentally right justified all signed numeric fields with the result that Purchase Orders were being received for quantities and values at one tenth their original worth!

12. Live Running

- CONTRACTS SIGNED?

To protect you and your Trading Partner you should insist that the written contract covering the points of your electronic trading is executed before you commence live running. In the event that there are practical difficulties, you should have a written agreement in place that will limit any liability in the event of dissent over what and when information was passed.

- MONITOR, AUDIT & CONTROL

Continue to monitor the facilities that you have put in place and seek to improve them at every opportunity. Do spot checks on data to ensure that the Control mechanisms that you use are providing you with enough information to satisfy your Auditors and Revenue authorities.

- REVIEW BENEFITS

Do not assume that because you have implemented EDI that you are getting all the benefits that you can. What other Documents could usefully be electronically conveyed to your Trading Partners, where can you obtain further improvements in process flows and staff utilization?

- REVIEW COSTS

Have you really made any savings? In this presentation I have assumed that you will but if you have not selected those documents or Partners that give you the greatest return, now may be a time to look for them. Also, you may have elected to use a PC based solution or indeed have commissioned the VAN to provide the translation service (and there now may be other VAN's serving your market area). Look at the costs and see if there are new and better priced alternatives. [In 1985 there were very few companies servicing the needs of EDI practitioners in the USA, now there are in excess of six hundred hopefully with better solutions available.]

PROJECT PLAN

13. More Partners

Having invested the time and money to get this far, take advantage of the facilities that you have in place and

SPREAD THE WORD

The more Business Partners that you get to work with you in EDI, the more potential cost savings you could make. There may come a time when the volume of Business with given Partners may not justify the full implementation of EDI, there will be a time where a company who is unable to respond to its Customers by providing an EDI service will find that those Customers will go elsewhere.

WHATS SO GREAT ABOUT EDI

Having looked at a number of the implications of using EDI, its worth considering one or two more to reinforce the benefits that can be achieved:

MUTUAL BENEFITS

So far the examples provided have only looked at one side of the Business Usage of EDI. Another example, that comes to mind is the situation where one company was sending an average of 40,000 Invoices a month to one Customer. It was taking anything up to 120 days to pay the invoices. The Customer was also having to use a data entry group of eleven people to key the invoices in, upon receipt - with its associated delays and errors. EDI seemed an ideal way of improving processing for both parties.

Each month, the full detail of the 40,000 invoices is passed by EDI to the Customer with four batch totals (about 10,000 Invoices per Batch). Four checks for payment are raised within three/four working days plus/minus any adjustments from the previous month. The Customer modified its existing Payables System to receive directly into it, the detail invoices and to automatically match against goods receipts. Most of the eleven staff were reallocated to other tasks within the Company, with those remaining concentrating on the mis-matches reported from the Payables System. The earlier payment was justified by the reduced cost of personnel resourcing, the creation of the EDI packages by the Seller, had its cost recovered by the improved cash-flow.

EFFECTIVE COMPETITION

There have been situations where not to use EDI has lost Companies Business. FORD in the USA declared that all suppliers would operate using EDI by the end of 1988. Those that were not prepared to no longer do business with Ford!

One US Corporation that supplied Hospitals throughout North America, although retaining its market value was losing ground in its market share. In 1947 they had about 40% of all Hospital Business but by 1984 they only had 18% (but still same value). They embarked upon a very aggressive strategy that involved the creation of a Network that all Hospitals could tie into with their existing systems (and where the hospitals did not have systems, simple systems were provided). Now, when a Ward Sister has a need for a bandage, she simply keys into a terminal and any understock level will automatically raise an order with the Network Supplier. Allied to excellent quality and comparable prices to their competition, the Network Supplier now has 45% of the market share and it grows each year. They are currently squeezing out their competition!

GOOD BUSINESS

The US Corporation that supplies Hospitals is an excellent example of strategic use of EDI.

Consider that about 80% of administration is spent on tracking the errors often generated by incorrect keying of data. Some organizations pride themselves on only having 2% of errors - but they still use an unrealistic amount of resource in correcting the problems. EDI will not remove errors that originate at source (if someone orders 10,000 instead of 100, EDI is not normally set up to catch that kind of problem), but it will remove the errors of rekeying.

Look at the Littlewoods example. Instead of just improving Customer satisfaction, they were able to release capital by selling no longer used warehouses - something that was not planned for in their early stages of EDI.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

SUMMARY

1. Allocate Resources

   If you want to do EDI right, make the People and Money available at the outset. By all means closely monitor where the time is being used and how the money is being spent. The investment will be repaid as many companies have found.

2. Do it with a Supplier First!

   Where possible, start doing EDI with one of your suppliers. Learn the problems in that relationship rather than with one of your better and major customers. Even with all the planning that has been suggested, you will encounter problems!

3. Be Prepared to Accept Change!

   Consider the example of Austin Rover where they no longer expect invoices from their suppliers. The Financial Controllers of many of their Suppliers took a long time in being satisfied that it was a better way to do Business.

4. TAKE ADVANTAGE OF EDI

   Look closely at your own organization. It has probably evolved based upon paper documents. Think of new ways to run your Business without relying on pieces of paper.

   In Europe, the Customs Authorities have created the Simplified Administration Document (SAD) that has replaced many types of forms. Part of the incentive for doing this was to help the introduction of EDI.

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

CONSIDER THE FOLLOWING

Just as an aid memoir to those readers considering embarking upon EDI it is worth answering the following questions:

- What does your Company do?

- Who are your Trading Partners?

- Which Transactions will you be using?

- Do you have any Special Interests?

- Do you belong to a Trade Association?

- Will you be using in-house, third party or off the shelf translation software?

- Will you be using Direct Connection or a VAN?

As a final note of caution

IT MAY COST MORE THAN YOU THINK

TUTORIAL - ELECTRONIC DATA INTERCHANGE (EDI)

THE ACTUAL ANSI X12 DATA MAPPING RULES

DESIGNING DECISION SUPPORT SYSTEMS USING X WINDOWS

by T C Ulrich

Mantix Systems Limited, 9 Progress Business Centre, Whittle Parkway, Slough, Berkshire SL1 6DQ, United Kingdom
Telephone: International: +44 628 668011

Introduction

Management decision making requires analysis of different views of information requiring time and skill to assemble in hard copy. The paper illustrates how an EIS product for managers in government and industry exploits the X Window system on HP 9000/300 series workstations to enable them to access and process information quickly without help from computer professionals.

Specific examples will focus on an Executive Information System for business managers controlling large and complex projects, but the same principles in system design could apply equally to general financial, marketing or production information systems which use a relational database. The common factor underlying all EIS applications is the need to display information in a very flexible and unpredictable way, and to permit a busy manager to achieve this with minimal learning. The process functionality will depend upon the manager's actual job, and is not the primary subject of this paper.

System Requirements

* a screen capable of presenting a number of graphs and text forms simultaneously for multiple views of the database.

* speed of response, and especially instant graphics.

* ease and flexible organisation of the screen as the project manager's desk top.

* ease of learning and use by managers who are too expensive and busy to spend time learning a complex system. It is particularly important that the user should not need to learn any programming or command language.

* a "layered" architecture which provides traceability and rapid retrieval of increasing levels of detail during problem analysis.

* a standard software environment, independent of hardware manufacturer.

* the ability to share data between managers.

* good data and access security.

* in larger systems the user should be able to access information from several databases over a communications network.

The X Window system meets all these requirements and it is surprising that the software industry as a whole has been slow to exploit this technology.

* it supports multiple users accessing multiple tasks on a LAN.

* summary and detail windows can be linked so the user can trade off window size, graphical format and screen space, in viewing the required detail.

* window management can be made simple for unskilled users.

* X menus are dynamically linked and need no expert mode like form-based menus.

* function parameters can be displayed as icons and selected by mouse.

* menus are tailored to the user's precise need and responsibility level. Learning time depends on user's "need to know".

* standard menus can be modified by user written processes.

* process access can be controlled with a simple fill-in-form technique.

Windowing Systems

The major contenders in the windowing software market are Microsoft (MS Windows), Sun (NEWS), Apple and X.

MS Windows was designed for PC's and when we started work did not support networked systems. NEWS and the Apple Macintosh both provide attractive options but are proprietary, and only X provides the required functionality and independence from a single hardware manufacturer.

Hardware Considerations. Workstations versus PC's

Simultaneous viewing of several graphs or forms requires a high resolution bit-mapped screen, as provided on the HP 9000/300 series workstation. Although there are a number of packages which run under MS DOS to make a PC into an X server, the low resolution of PC screens makes them generally unsatisfactory. Our experience has been that users who started with PC-based display servers quickly realised the added benefit of a larger high resolution screen and upgraded their systems to the pleasure of the HP salesman.

The other important benefit of using a workstation is the power, and in particular the high speed disk interface, which enables the application developer to provide good response even to complex enquiries.

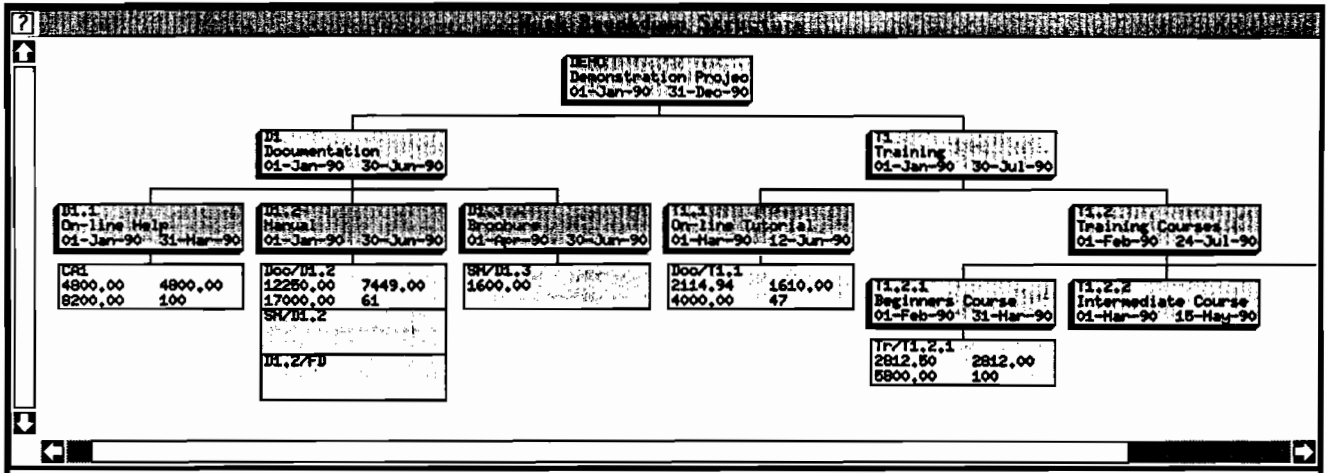
Screen size can also influence users into paying the extra cost of a workstation. A 19" screen has almost double the area of a typical PC screen and up to 5 times the number of pixels. However, regardless which screen size is chosen, **it is fixed** and optimising the use of this fixed area is essential.

Optimising Screen Layout

As with all windowing systems the user can move and size an X window on screen. Scroll bars denote whether the display is a partial or total view of the data passed from the client application. The user can also overlap, occlude, raise and lower windows so as to optimise use of fixed screen space. To extend his ability to trade off variable information against fixed screen space we were able to provide different display format for graphical elements and use sub-processes to create linked forms which contain detailed information. (Figure 1)

Multiple User Support

Since X operates in a multi-user multi-tasking networked environment it is possible for a manager simultaneously to view data belonging to different subordinate managers who are working on a distributed database or different aspects of the same data (Figure 2). This enables him to make instant performance comparisons and identify problem areas very rapidly. Also, because X supports full graphics the software developer can provide tremendous visibility for the executive and his staff. Such a system is most likely to find favour with organisations who delegate responsibility and encourage open discussion of problem areas.



WORK BREAKDOWN STRUCTURE with

Work elements (blue) showing:

- WBS Element Code
- Description
- Baselined Start Date
- Baselined Finish Date

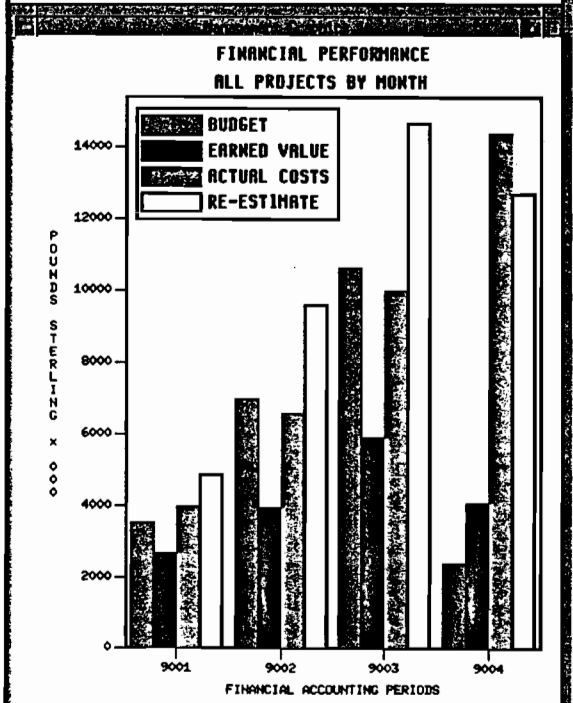
and Cost Accounts (orange) showing:

- Cost Account Code
- Budget (BCWS)
- Earned Value (BCWP)
- Actual Cost (ACWP)
- Percent Complete

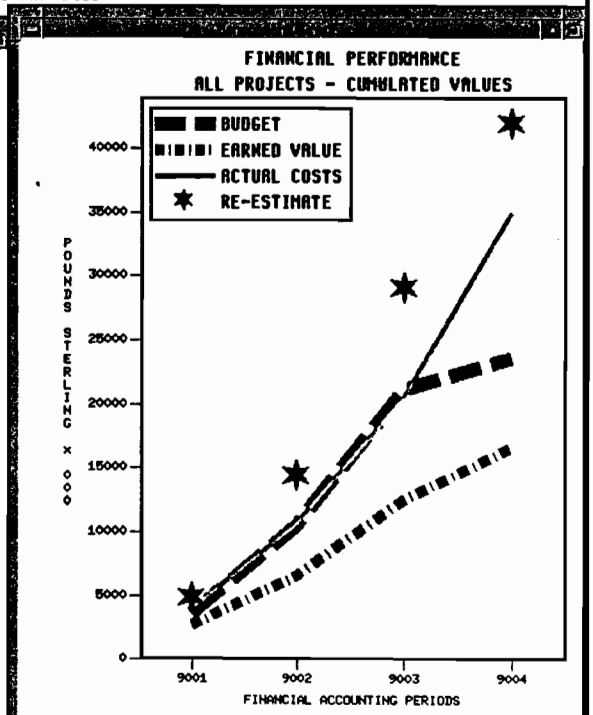
6011-3.1

Figure 1

Management reports showing consolidated financial information across all projects by month, and by cumulated values to date, and displayed in different formats.



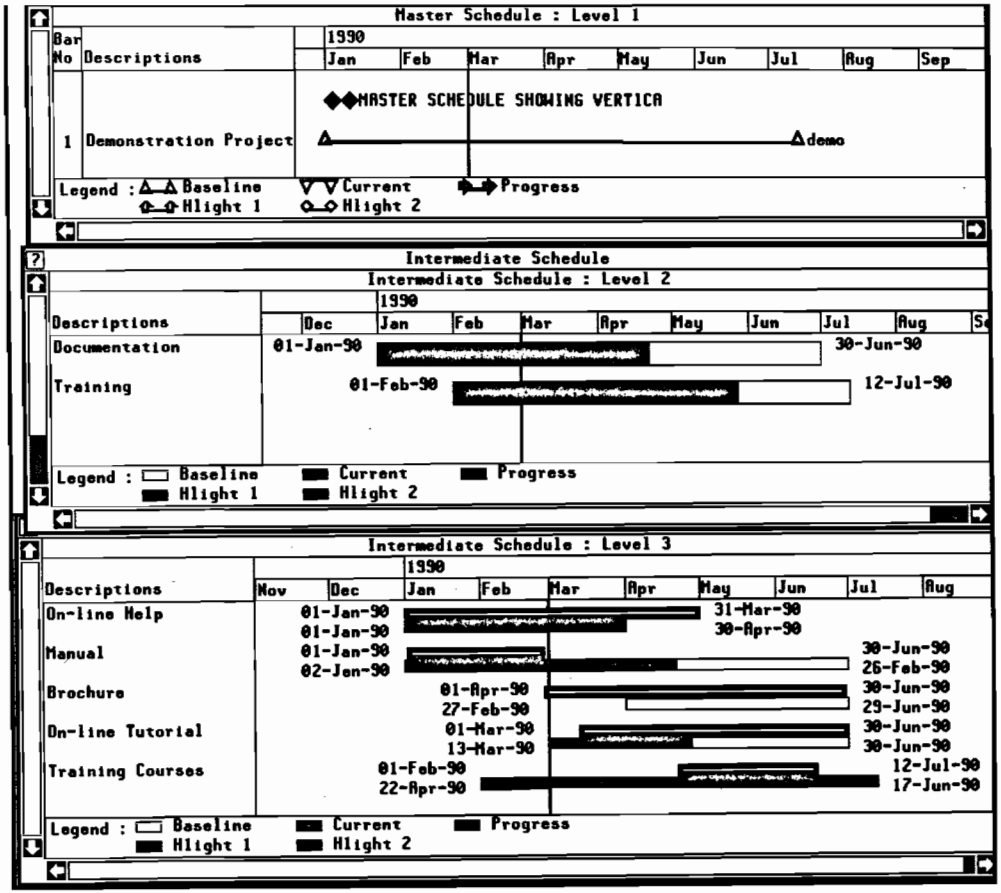
BUDGET	3526,680	6986,280	10632,160	2432,320
EARNED VALUE	2650,000	3972,000	5939,000	4110,000
ACTUAL COSTS	4000,000	6600,000	10000,000	14400,000
RE-ESTIMATE	4876,680	9614,280	14693,160	12722,320



BUDGET	3526,680	10512,960	21145,120	23577,440
EARNED VALUE	2650,000	6622,000	12561,000	16671,000
ACTUAL COSTS	4000,000	10600,000	20600,000	35000,000
RE-ESTIMATE	4876,680	14490,960	29184,120	41906,440

6011-3.2

Figure 2



6011-3.3

Figure 3

Using Windows to Trace from Summary to Detail

Figure 3 shows how with 3 windows on screen simultaneously a project manager can trace the progress of various intermediate levels of the schedule against the master schedule. On paper, space does not permit more than 3 windows, but on screen there is no practical limit to the depth of detail which can be examined by overlapping windows. Manipulating window priority to obtain instant graphics is easy.

Ease of Learning and Use

X applications can be designed to be entirely mouse-driven. The only typing required is for data values and text. The menus are structured and lower level menus are indicated by an arrow to the right of any entry. They can be dynamically linked when the user moves the mouse cursor to the arrow. (See figure 4).

Once the user understands the menu structures he need only gain minimal dexterity in mouse operation to become effective. Since accessing lower level menu processes is a single operation there is no need for an expert mode or command language to overcome frustration at repeated menu operations. The seamless join between novice and expert is defined by the skill with which he uses the mouse.

Rules for mouse operation are also very simple. The right button brings up a context dependent menu and the left button denotes actions. By careful design the action can be intuitive and employing consistent terminology helped the user to learn quickly. The action button can be used to display details about a graphical element, to add, modify or delete an element, and to manipulate icons on forms.

Process and Data security

Ease of learning can be enhance by using the security system to configure user menus on a "need to know" basis.

Figure 5 shows how a Cost Account Manager can be inhibited from changing the Work Breakdown Structure or Work Scope (WBS dictionary) by simply removing these functions from this menu set. Each permission screen is a separate X Window and can be easily manipulated by a security specialist who need have no special computer expertise.

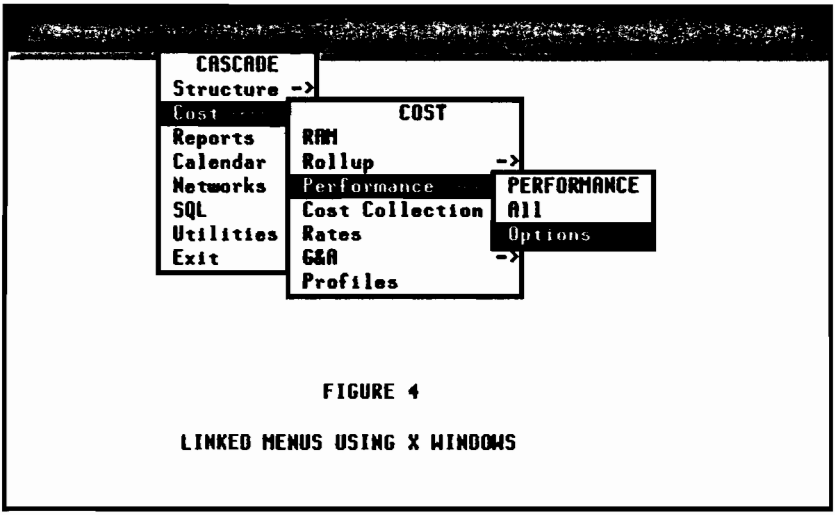


FIGURE 4

LINKED MENUS USING X WINDOWS

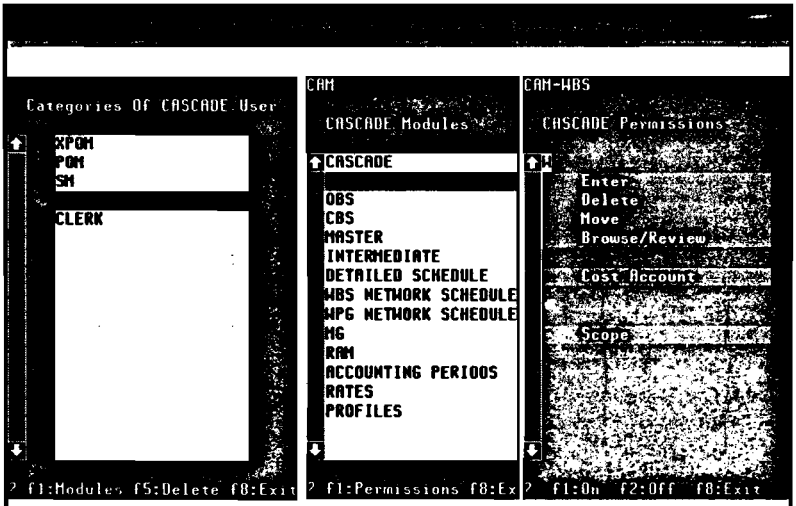


FIGURE 5

PERMISSION SWITCHES FOR MENU CONFIGURATION

Customisation

It is easy for a software developer to build hooks for the user to add his own processes as menu options. Having written and checked out his process the user need only write a simple UNIX shell script which is invoked by the menu entry, which itself invokes the user process. Control can be returned immediately to the window manager if the process can run in background or on termination if the user wishes to inhibit use of the workstation during process execution.

Operating System Considerations

Our application is unusual in that it uses X Windows with the ORACLE relational database. In order to achieve interactive performance we found it necessary to build a database cache between the application and ORACLE. As a result we hit the limit of 60 file descriptors per HP-UX process and this caused some problems. These were solved by recycling file descriptors which had been timed out and some experimentation was necessary to determine the best way of handling this.

In a distributed environment TCP/IP uses one file descriptor per connection to the database server. This is aggravated by the fact that the typical X Window user is running several simultaneous processes. In consequence, on larger systems it is possible for users to be locked out even though the hardware would in theory support them.

It would be highly desirable if HP-UX could be improved so that the number of file descriptors per process were configurable as is the case with VMS.

Conclusion

The X Window environment is so well suited to decision support applications such as project management or Executive Information Systems that we should expect to see a range of products which exploit its capability over the coming years.

HOW TO MEASURE PRODUCTIVITY

A Technical Answer to a Management Problem

Ronald E. Stanton, CDP

RES Software Products
P.O. Box 66
Fairport, New York 14450-0066
United States of America

Computer Education Learning Center
P.O. Box 66
1276 Fairport Road
Fairport, New York 14450
United States of America

INTRODUCTION

One of the most frustrating problems faced by data processing management is to find a way of measuring the productivity of technical personnel. Whether one reads the so-called miracle of the Manhatten Project or more recent articles found in the trade journals, management has been unable to come up with a standard methodology. From the nightmare of '10 lines of debugged code per day' to the claims that programmers will not be needed within five years, we are constantly battered with theories and quality circles. Without ignoring the fact that programming is a true art and requires a creative mindset, we tend to overlook the customer's perspective. We also seem to ignore 'imposed' deadlines and 'off-the-wall' estimates that are presented without full consideration for the complexity of technical development.

Every time we hear the phrase 'All you have to do is.....' a silent alarm should go off in our heads. Rationalization can seldom replace the wisdom of objectivity. After a quarter of a century of frustration and turmoil, we should really use the skills we developed to attack this nagging problem. We should attempt to create a technical solution for the problem by working with facts and relating them to a communications dilemma between the data processors and the user community.

GETTING THE FACTS STRAIGHT

Can any of us remember the number of times productivity has has been brought up in planning sessions and staff meetings? We always seem to have plenty of questions about it and very few explicit answers to offer. We seem to be caught without enough facts to back up our assumptions and end up agreeing that we need to do something about it. As data processing professionals, we deal with data as a commodity and often manage to parlay some of our ideas into resources and system upgrades. We seldom manage to utilize our basic talents to effectively market the MIS Department.

As managers, we are constantly trying to find enough time to get to those things that we feel will be able to quantify our productivity. Regardless of the reasons, meetings, reviews, or any other unplanned activity, we don't seem to get at it. Considering the number of intrusions into the normal workday, it is almost impossible to re-arrange our schedule and act on those 'nice-to-have' items.

Structured methodologies stress the need to evaluate current environmental circumstances in order to produce an effective design. This step is often skipped in companies where the management mentality relates to getting started instead of getting correctly started. If we evaluate this approach and relate it to productivity, we hear a 'work smarter' analogy.

In summary, we need to understand the management style within our organization before we can attempt to resolve any type of productivity measurement questions. If our management tunes in to negative comments instead of positive ones, the problem becomes a double-edged sword where education and productivity are on opposite sides. The trick here is to educate managers to present materials in a positive way in order to reduce the amount of time spent on complaining. In other words, present a positive approach that will logically identify production figures for the MIS Department. We should also measure these numbers from period to period for trending and the potential isolation of specific control factors.

Soliciting involvement from users in identifying what they need to know about outstanding projects and maintenance tasks would be an outstanding way to start. Marketing a required product is always easier than trying to identify a need for a product. These needs could then be fed back to the users in the form of a review document and finalized based on their feedback. This approach would also convey a positive note on a customer sensitivity.

UNDERSTANDING WHAT WE NEED TO KNOW

If we analyze a business function with a structured approach, we tend to cover the realms of:

- What did we do in the past?
- What do we think we are doing currently?
- What are we really doing currently?
- What do we want to do in the future?

As a service organization, data processors should relate the questions above to reporting progress in lieu of defending a philosophy of productivity measurement. From the customer's perspective, you are either delivering or you're not. You have either made progress since the last report or you are falling behind schedule. All we have to do for the customer is provide a logical summary of activity for their department and keep them informed on a regular basis. If we are honest about measuring our own activity, we can easily translate the data into a decision support status format.

The true acid-test is to present the 'bottom-line' picture to each user department and a summarized version that will show what data processing has accomplished for the total company. Unless we are willing to stand tall and honestly measure our accomplishments, we will have a difficult time convincing the user community or management that we can actually do it.

TELLING IT LIKE IT IS

In order to report progress one needs to have a prior status to report against. For data processing, this would be a count of all open requests from the prior reporting period. This number would have to be broken down into several columns covering the number of open requests, the estimated resources for the open requests, the number of active requests, and the estimated workdays for the active requests.

With the baseline established, we can now address all of the activity that affects it. If we relate this activity to order entry processing, new orders, order cancellations, the completed orders, and order modifications would have to be processed against the previous backlog. This means that we would add columns for requests received, cancellations, items completed, and the workdays for each of the activity areas. Modifications would normally affect the estimated workload and not the number of projects. One additional column would be required; one containing the number of items that activity was initiated for during the current reporting period along with the estimated workdays for the new items.

With the baseline established and the activity against the baseline being tallied, we can now produce columns that will tell us the net results and the entries that will appear in the baseline for the next reporting period.

WHAT HAVE WE DONE TO OURSELVES?

Presentation of the information defined in the prior sections of this paper as-is could spell potential disaster for a D.P. Manager. In covering all of the activities that took place for user requests, we omitted discussions of the data processing department servicing itself and other routine maintenance performed on a regular basis without accurate documentation and tracking. Items like vacation, holidays, illness, training/education, and administrative activities that can take up a major portion of our available time were left out.

When it comes to the question of just how productive we are as a group, we have to tell it like it really is; including the non-productive time spent on miscellaneous administrative tasks. For most organizations, the maximum potential percent of available time for customer service is less than 60%. In some shops, it can be less than 40%; due to extremely high system maintenance for production failures and poor quality system documentation.

Consider the following illustration:

- Total maximum number of workdays in a year.... 260

  -- Company Holidays..... 12
  -- Earned Vacation..... 15
  -- Illness..... 8
  -- Travel Time..... 5
  -- Personal Time..... 3
  -- Training/Education..... 7
  -- OJT/Orientation..... 10

  Total..... 60

- Remaining available workdays..... 200

  (Approximately 77% of the total available time)

UNDERSTANDING THE IMPLICATIONS OF AVAILABLEE TIME

If we were to schedule a person for a 20 resource day task, we would normally look at the calendar for upcoming holidays and possibly check the department vacation schedules. With the facts in hand, we would subtract the available workdays in the next two months until we reduced the estimate to zero. We can almost automatically assume that the resource days were equated to workdays on a one-for-one basis. This way, the person assigned the task would be scheduled to complete the job in approximately 5 calendar weeks. Also, we may neglect to consider what other open items that person may be currently working on. We would inform the user area that the project was scheduled for completion on or about a specified date and target another open item for the same type of scheduling exercise.

If we really listened to what the facts were telling us about productivity, we would go through the following analysis:

- Consider an efficiency rate of 85% for the 200 remaining available workdays..... 170

- Out of a total of 260 potential workdays in a workyear, we can expect approximately 170 workdays of production from average employees.

- The maximum produotivity level for an average employee in a data processing position would be approximately 65%.

- The number of potential workdays against project estimates would be only 14 days out of a 23 day month.

- At the above rate, it would take an average programmer approximately 7 weeks to complete a 20 day project; assuming minimal or no intrusions into the work schedule. Also, assuming that the estimate was accurate.

Some of us might say that we were allowing for the efficiency level more than once. This wouldn't be true considering the fact that we really derived a productivity ratio of 65% for an average employee. Using this efficiency ratio, we could divide the estimate by 0.65 and adjust the estimate for use against calendarized workdays. This calculation yields an extended estimate of 31 calendar-workdays (rounded).

Whether we believe the previous examples or not, we all realize that the data processing industry appears to be poor at estimating projects. Considering the efficiency theory for scheduling people only 14 days per workmonth, we could test the theory by logging only the actual workhours for an open item. This way, we could analyze the relationship between the logged hours versus the calendarized days taken to complete our tasks.

We have also established a basic minimum productivity level for our employees to be measured upon. The more accurate our estimates, the more valid the productivity level becomes. By tracking how we are doing on our estimates, we would be able to adjust them by user area or system in order to adjust the productivity level for our own organization. All these steps will take time and energy to support. If we don't try to do it, we may never be able to tell a user when a job will get done. Worse yet, our immediate manager may be tracking the productivity levels more accurately than we are. If we have facts to present, it will inform management that we are truly in charge of our group. They will also realize that we are attempting to measure ourselves with a goal of improving the productivity levels of our employees.

Historical tracking of project progress for estimates and job completions would provide a tremendous amount of input for an MBO analysis. It would also provide a complete snapshot of how the group delivered against plan and a profile of what user areas received the most services. In summary, it makes sense to track our own groups. If we are tracking ourselves and make an honest attempt to communicate this fact in status reports, management will normally respond accordingly.

COMMUNICATING THE FACTS

Considering the prior example of productivity/availability, we would likely meet a stone wall of resistance if we tried to sell the management community our analysis. Instead, we should try to document the facts with the status reporting and let management draw their own conclusions about the available development time. They will soon realize that it will take a department of 10 to accomplish the implied workload for a group of 6 people. If we have a department of 3, we can only produce the equivalent of 2 people.

If we look at our cliche's that tell us it always takes twice as long as our estimates, maybe our estimate was pretty good and we just had to adjust it for the maximum productivity potential of 65% and we would get the project completed near schedule. By dividing .65 into a 20 day estimate, we will come up to a 31 resource day estimate. Equating 1 resource day to a weekday, the project will take about 7 weeks to complete. If we selected the 14 day per month scheduling approach, we would come up with a 32 calendarized workday schedule. Either way, we will be allowing for most of the problems that we have to deal with on a daily basis.

The only thing that we did not consider in our productivity estimate was the impact of having to stop what we are doing to answer routine questions and address other daily chores. The more interruptions that occur, the less efficient we are. As a rule-of-thumb, some systems professionals add on 2 days to a schedule for every 8 hours of distractions. If we want to control the delivery dates for projects, we should try to reduce the number of times our personnel are called away from their main purpose--project development. Without this type of control, we may never be able to keep up with our development schedules.

APPLYING WHAT WE HAVE LEARNED

When the question of productivity is addressed we normally think in terms of a single employee. Questions about why one person only seems to produce half as much as another person or why one system requires five times as much time to change versus another system always seem to be battered around.

A thorough analysis of our activities over a period of at least 6 months will begin to provide some of the answers we are groping for. This will require us to make a concerted effort to estimate, reasonably schedule, and track our work tasks. By taking a hard look at the department/group as a whole, we may gain a lot of insight on how to fairly evaluate individual employees. We have to know our easy systems and our hard systems. We also need to track the number of tasks we are required to log and estimate on a periodic basis. All of this information along with the tracking of the tabu' area of 'non-productive' time will enable us to understand our organization and adjust to meet its' needs. Non-productive time should be placed into our tracking system as 'Misc' or some other un-alarming name. Tact is really necessary when we begin to identify and document the fact that we have time that is not really applied to projects.

The critical factor in being able to measure the productivity of individual employees is having a thorough understanding of the total environment. Without this knowledge, we will tend to base our evaluations on subjective assumptions instead of factual evidence. If we need to turn on the pressure to improve productivity, we should apply the pressure across all lines. Since we have developed a productivity standard, we can convey this fact to the technical personnel and let them know how we will be measuring them in the future. In most cases, if you let a person know what measurement tool you will be using, they seem to adjust rapidly and begin to bombard you with results. Measure someone on status reports and they will normally spend a lot of time developing the reports and they provide elaborate verbage for each and every item on their report. If you tell them you will measure them on their productivity against the department standard, you better be willing to do more for the more productive people or they will quickly lose interest in your goals. It might even be better if you involve the employees in the development of the standard so that they have ownership in the process. Also, if they create the yardstick, they better be willing to be measured by it.

IN CONCLUSION

Over the past thirty years, MIS management has been actively wrestling with the topic of productivity. Instead of using their finely-tuned skillset to develop a simple and effective solution, they have yearned for help instead of earning it. Some eager vendors have been able to make tremendous profits feeding on this implied need. There are even a myriad of packaged software systems available for PC's. No matter what system a person purchases, they will have to change many of their processes in order to adapt to the generic packages. Some groups even have several packages being used at the same time.

Instead of looking for a total solution, we may be better off with a functional solution. By tracking individual projects in a manual mode or with a simple system written in a 4-GL, we can first attempt to measure the productivity of a group. Since statistics normally seem to hold true for predicting election results and football bets, we could then utilize the statistics to measure teams or even individual employees.

The development of a simple system that could track project activities from initiation to completion would enable one to provide a bottom-line summary of what projects were completed and the activities logged against active projects. The project details for each user area could be distributed as a status summary and reduce the amount of verbage required for monthly reporting. The department/group summary could be used for analysis of project estimates and the productivity measurement standard. At year-end, a month by month summary could be generated to portray the productivity trend and the gross tally of projects completed. The work-in-process could also be viewed in order to establish the fact that until the tasks are reported complete, they will only appear as time logged against active projects.

In summary, technical solutions to productivity measurement are possible. They will be as individual as the shop they are created for and may only be designed to address specific issues. No matter how they are designed, they will need to be used by open-minded managers who will tell it like it is.

TITLE: Software Development - The Off Shore Option

AUTHOR: Vik Shroff & Duane Percox (408) 937-7814
Blue Star Group
19925 Stevens Creek Blvd., #119
Cupertino, CA 95014

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 6013

TITLE: HP Strategy Sessions

#7001 - 7009

Strategy sessions only.

AUTHOR: For more information, contact:

Nancy Onskt, Hewlett-Packard Co.

19091 Pruneridge Ave., MS-46LK

Cupertino, CA 95014

FINAL PAPER WAS NOT AVAILABLE AT TIME OF PRINTING

PAPER NO. 7001-7009