TESTING A COMPLEX, REAL-TIME MANAGEMENT INFORMATION SYSTEM

By

WARREN DANIEL NILSSON

Bachelor of Science
University of Rhode Island
Kingston, Rhode Island
1964

Master of Science in Engineering
New York University
Bronx, New York
1965

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
DOCTOR OF PHILOSOPHY
July, 1973

TESTING A COMPLEX, REAL-TIME MANAGEMENT

INFORMATION SYSTEM

Thesis Approved:

Hamed K. Eldin
Thesis Adviser

S K Adams

Fred Black

Earl J Ferguson

D D Durham
Dean of the Graduate College

PREFACE

A successful, large, real-time management information system just does not happen. There are many challenging constraints which must be properly considered or disaster may result. Its development, test, and installation activities must be planned and controlled. There has been especially limited treatment of the testing aspects, but it is here where the most serious schedule and resource commitments have been previously missed. This dissertation develops a general model of the eight phases assumed to constitute the management information system cycle with special emphasis on the testing activities. Then, the general model is applied to a specific project. It demonstrates that it is possible to develop a general management information system model which can be successfully applied to a specific management information system. By utilizing the approach provided in the dissertation, it is possible to meet the testing objective - there must be a structured, non-redundant, series of progressive tests which insure that the management information system will be installed in the shortest amount of elapsed time with a minimum resource level.

I work for International Business Machines (IBM) at Sterling Forest, New York. My present assignment is to coordinate the various activities of a complex, real-time manufacturing information system (MIS) which is to be installed in IBM locations throughout the world. Special emphasis of the work assignment is placed on the testing aspects which insures that the user requirements are incorporated in the management information system. My advisory committee approved my proposal to study this area.

This dissertation is the culmination of a Ph.D. program. It was undertaken with the support of an education leave of absence from IBM for the entire 1972 calendar year. The study for the dissertation was begun in 1971 and completed in 1973. It relies heavily upon my previous work experience at IBM during the past eight years. The opportunity provided by various work assignments and also the educational leave of absence is greatly appreciated.

This dissertation consists of six chapters. The Introduction, Chapter I, provides a background for understanding the historical development of information processing and then discusses how computers are used in organizations. The Problem Formulation, Chapter II, is where the problem is stated, the purpose of the study is explained, the literature search is examined, and the procedure or methodology used to insure the purpose is provided. Chapter III provides a discussion of Management Information Systems. It first defines the term "Management Information Systems" by quoting various definitions and then by developing the definition used in the remainder of the dissertation. Then, the eight phases assumed to constitute the MIS are examined, the advantages and disadvantages of real-time MIS versus batch MIS are discussed, an approach to a MIS organization is developed, and a planning and control methodology which is to be used throughout the remainder of the dissertation is provided. In Chapter IV, the general model for a MIS is developed with the emphasis on the testing aspects. The chapter begins by discussing the various advantages and disadvantages of modeling. It proceeds by providing the model considerations and then discussing the type of computer languages which should be considered to meet the model considerations. It then develops the rationale for selecting MINIPERT as the computer language used in this dissertation. Finally, the general model is developed and described according to the planning and control concepts provided in Chapter III. It must be emphasized that the eight phases of the MIS are modeled to provide the framework but that most of the emphasis is on the testing aspects. Chapter V begins by describing the actual MIS. It continues by describing the application of the general model to the actual situation. Then, the model analysis and validation procedures are discussed. Chapter VI is the last chpater and it provides an overview of the entire dissertation. It provides the summary, conclusions, and recommendations. If one desires an overall understanding of the dissertation, it is suggested that the last chapter be read.

I wish to express my appreciation to the members of my advisory committee: Dr. S. Keith Adams, Dr. Hamid K. Eldin, Dr. Earl J. Ferguson, School of Industrial Engineering and Management, and Professor Fred M. Black, Administrative Science Department. The committee assisted me in the planning of a program of study, selection of a research area, and in the major portion of preparatory course work. Special appreciation is expressed to my Major Adviser and Thesis Adviser, Dr. Hamid K. Eldin, who provided not only the initial guidance but greatly assisted in the research phase.

There have been others who have contributed to the successful completion of this study: Mr. John B. Hudak, IBM Manager in Common Systems Development, who has provided the continual keen insight into various information system problems; Dr. Philip E. Hicks, Chairman of the

Department of Industrial Engineering, New Mexico State University, for his motivational influences; Mr. Martin F. Ziac, IBM Director of Common Systems Development, who has developed and has implemented a successful strategy from a quagmire of squalidity; and many others.

Finally, special thanks go to my parents, Warren O. and Anna G., for their continual love, understanding, and confidence.

TABLE OF CONTENTS

Chapter	Page

LIST OF TABLES

LIST OF FIGURES

Figure | Page

CHAPTER I

INTRODUCTION

Information Processing Revolution

For centuries man did not quantify information. He began to count on his fingers and toes only because of the need to survive. But, when it came to such things as counting all the cattle in a field he was in a quandary, so he started to use pebbles, notched sticks, and even knotted ropes. Next, he progressed to the point where a computational aid was needed to keep track of the many tedious arithmetic calculations. One of the first items utilized was the abacus. This device has lasted as the most important computational aid over the past 2000 years. In fact, today, it is probably the world's most widely used calculator.

The next major development occurred when man progressed from the manual stage to the machine assisted stage. This began in 1642 when Blaise Pascal introduced the first adding machine. Three decades later, a machine that could add, divide, subtract, multiply, and extract roots was invented by Gottfried Leibniz. Then, in the 1820's and early 1830's, Professor Charles Babbage, generally regarded as the godfather of the digital computer, conceived an "Analytical Engine". Many people consider this to be the predecessor of the automatic computer. It had a unit called a "store", which was a memory, and a data processing unit or arithmetic unit called a "mill". Data could be entered by either setting the dials or by the use of cards. When there were intermediate results, they were placed in the "store" and obtained later as needed. It was an information machine designed to be capable of performing complex processes on information without manual intervention. It was designed to have a 20 place accuracy. However, due to the lack of technological advances in this time period, the machine never became functional. But, it did establish the basic computer concepts.

The next major improvement occurred in the late 1930's and early 1940's when the Automatic Sequence Controlled Calculator, MARK I, became the first programmed computer to operate successfully. It was developed and built at Harvard University by Professor Howard Aiken with the financial and technical support of International Business Machines (IBM). This electro-mechanical machine consisted of punching, reading, and printing units along with the appropriate cams, counters, relays, etc.

A significant breakthrough in engineering technology took place when the Electronic Numerical Integrator and Calculator (ENIAC) was developed - the world's first electronic computer. It was designed by Dr. Presper Eckert and Dr. John Mauchly at the University of Pennsylvania in cooperation with Major Herman Goldstine of the U.S. Army. It was designed to help in the calculation of mathematical tables for the firing of projectiles. Its approximately 18,500 tubes and 1,500 relays all had to work correctly to obtain valid results.

Basically, a computer consists of two parts - the hardware and the software (1) which is portrayed in Figure 1.

The hardware consists of memory; physical entities such as the input devices, output devices; and the central processing unit (which contains the control unit and arithmetic unit). The programs and their related data are fed into the memory through the input device. Then, the instructions are sent to the control unit which, in turn, sends some data to the arithmetic unit and also tells the arithmetic unit what to do. The arithmetic unit contains a section called the accumulator where various arithmetic operations are performed and the results are saved. After all the instructions are completed, the results are sent to the output device where the human utilizes the information.

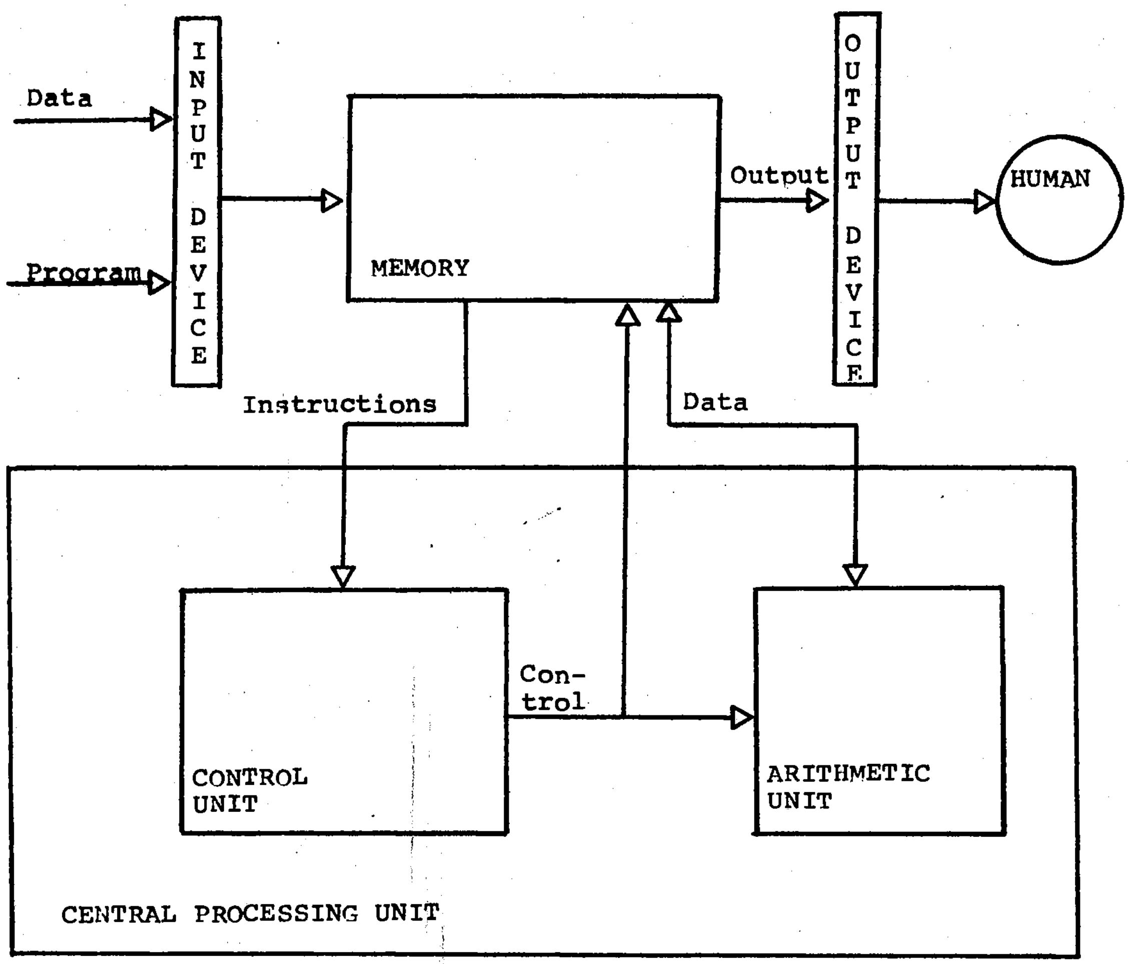


Figure 1. Computer Hardware and Software

The first generation of computers, from approximately 1954 to 1959, were generally developed for scientific uses. The major emphasis was placed on computational capability, rather than ease of input or output. The vacuum tube was the essential ingredient and also the main problem (due to relatively short life, bulk, and heat). The second generation of computers began with the introduction of the transistor. These computers had small, magnetized rings or cores for storing data and instructions, greater computing power, and were smaller and faster. The third generation of computers began in about 1964. Its main distinctive factor was monolithic circuitry. These computers provide even greater processing speeds, and enhanced input and output performance. Also, the third generation includes the development of advanced programming concepts (operating system) which allows the computer to control the handling of jobs on a priority basis, checking for input and output data errors, job accounting, and corrective action when errors are found. Currently, we are entering the fourth generation of computers denoted by the extension of the micro electronic concepts into such areas as Large Scale Integration (LSI). This type of extension makes it possible to greatly increase the number of circuits per square foot. In the foreseeable future, a computer equivalent to the System/360 Model 50 will be put together on an 8-by-10 inch circuit board. By increasing the computer circuit densities, one is not creating a revolution, such as the one when the transistor replaced the vacuum tube, but rather an ever increasing evolution.

The computer can do many powerful things, but at the same time it has some inherent limitations. First of all, computers are not inexpensive. Even though the average cost of computation has been steadily decreasing, a substantial investment is required. Also, the computer, contrary to what many believe, does not possess any intelligence. It cannot solve problems for which it has not been programmed. However, there are some real advantages which will probably offset the limitations. For instance, the computer can perform precisely and quickly. The time to perform calculations is measured in nanoseconds (billionths of a second) with precision going to sixteen decimal digits. It can work for thousands of hours without errors on the most trivial or complex problems. It can also store huge quantities of information in an integrated data base which can be retrieved by many different users in seconds.

There does not seem to be a visible stopping point for the information processing revolution. Computers are being utilized for more diverse activities. In banking, there will be a money-card system where cash will not change hands. The stores will have terminals which will be utilized to debit and credit the customer's account on the central computer. If there is not enough money, the system will respond by either rejecting the customer's order or perhaps advancing credit for an appropriate charge. In teaching, there will be greater use of computer-assisted instruction. The student will progress at his own learning rate. The computer will have parameters on the student's score and learning rate, and when necessary will provide more individualized drill. In business, there will be common integrated data bases which will provide stronger ties between the logistics, personnel, and financial areas. The total systems approach to managing will be much closer. There will also be major advancements in the useage of computers in many other disciplines, such as medicine, police, simulation, etc.

An excellent summary of some events which will further enhance the information processing revolution are listed in Table I. The projection was made in January 1969 by G. B. Bernstein and was published by Rothman and Mosman (1).

As can be seen in Table I, the computer, if planned correctly, can help insure more individuality by being able to reduce conformity. Ideas, instead of being grouped and then tested in mass, can now be individually considered. Manufactured products, such as automobiles, can be made to customer request by utilizing computer scheduling techniques. Hopefully, in the long trend, the individual will benefit.

The information revolution, however, has not been a smooth process. There is a myriad of problems when attempting to develop, test, and install computer programs. Today, there are many challenging areas which need to be investigated so that it is possible to implement computer programs in the shortest elapsed time with the least amount of resources. Problems become intensified during the testing activities where the resources are always the highest and the schedule is usually the tightest. It is for these reasons that there is a need for further investigation.

Computers in Organizations

The initial application of computers was in scientific and engineering applications. During the past two decades, organizations have started to use the computers for business applications. Usually the accounting function is the first candidate for computerization because

TABLE I

FIFTEEN-YEAR FORECAST OF INFORMATION TECHNOLOGY
(Source No. 1)

Event	Reasonable Change of Occurring In (Probability 20%)	Expected To Occur In (Probability 50%)	Almost Certain To Occur In (Probability 90%)
HARDWARE			
Communicating With a Computer			
English -- limited	1974	1980	1990
English -- good	1980	1985	1995
Voice -- limited	1971	1974	1978
Voice -- good	1971	1980	1987
Optical Character Reader			
10K char/sec	1972	1977	1981
cost < $20K	1970	1972	1974
script	1975	1980	2000
Automatic Fingerprint Recognition	1986	1990	2000
Speech Recognition to Identify Speakers	1973	1977	1983
High Density Electro-optical Storage			
5-20K bits/square inch	1970	1973	1976
200-fold Improvement in Cost/Operation	1974	1978	1985
Memory Cost			
<1 sec access	1971	1974	1977
.001 mil[a]/bit			

TABLE I (CONTINUED)

Event	Reasonable Change of Occurring In (Probability 20%)	Expected To Occur In (Probability 50%)	Almost Certain To Occur In (Probability 90%)
Memory Cost (Continued)			
10^{10}-10^{12} bits			
read only			
Mass Storage			
10^{11}-10^{12} bits	1972	1975	1978
< 1 sec access			
< 1 mil/bit			
10^{7}-10^{8} bits	1970	1972	1974
< 1 microsecond access			
1 mil/bit			
Laser-Oriented Memory and File Storage			
10^{9}-10^{10} bits	1973	1976	1979
few mils/bits			
Erasable Mass Storage			
10^{10} bits	1973	1976	1979
< 10 millisecond access			
.1 mil/bit			
10^{13} bits	1974	1977	1980
< 1 sec access			
cost .1 mil/bit			
10^{15} bit memories	1975	1980	1982
10^{-7}¢/bit			

TABLE I (CONTINUED)

Event	Reasonable Change of Occurring In (Probability 20%)	Expected To Occur In (Probability 50%)	Almost Certain To Occur In (Probability 90%)
SOFTWARE			
Machine Indexing of Text and Picture Data	1972	1976	1986
Natural English for File Inquiry	1974	1976	1985
Cheap, "accurate" Indexing of Natural Language Text	1972	1974	1976
Accept and Learn the Semantics of Natural Language	1971	1973	1975

it possesses high volume and standarized clerical and statistical data manipulation. As a result, payroll, accounts payable, and accounts receivable are the first jobs to be adapted to the computer.

Today there are over 50,000 computers being utilized throughout the world. If organizations are to meet the changing environment and competition, more imaginative computer applications must be devised. The explosive growth of computer technology manifesting itself in improved reliability, speed, and capacity has made it economically advantageous to apply the computer to previously unconsidered applications. It can be realistically assumed that computers are capable of doing almost any job where the limitations seem to be solely on the imagination of the users.

During the twentieth century, there has been enormous growth in the size of organizations. This expansion has generally produced decentralized divisions which have their own staff functions. These functions are generally duplicated within some of the organization's other divisions. There was a need for each division to have an individual computer system which would handle the division's unique problems. This creates a hardship on the information system throughout the entire organization. Each function usually has its own operating procedures which can cause problems in attempting to derive common information for the entire organization. It makes the decision-making process more complex and longer than desired. With this in mind, it can be quickly seen that there would be different perspectives, dependent upon one's location in the organization, on the question of centralization versus decentralization for both the computer system and the planning and control functions. In the past, each division has generally had its own computer system which was tailored to its own needs. However, if the computer system is centralized, there are some obvious advantages and also some hidden disadvantages. Technically, it is not a problem to have one centralized computer system. The formation of an integrated data bank provides common information to everyone in the entire organization and thereby to every staff function in each division. It is possible to treat information requests on an overall organization basis rather than differently at each division. For instance, a large multi-plant manufacturing organization would be able to purchase all materials utilizing a centralized computer system. This would result in an economy of scale. Another advantage is that one large central computer would offer lower cost per unit of processing. This would result in not only greater operational efficiency, but probably also in more varied services. Therefore, it would be easier to implement new company programs and also to attract more key professional personnel. Other significant savings would result from only one common data preparation step for all personnel, from having only one data base which is commonly understood by all employees, and from having one system which can react to change. Also, it would be possible to transfer products, programs, etc. from one division to another. One of the most important benefits occurs when a production schedule is changed. It could take as long as three months to determine the new detailed requirements using batch processing and manual methods at different locations. Utilizing a centralized computer system, it would be possible to shorten this entire cycle to three weeks. But, the case for decentralization of the computer system also has some significant benefits. The larger the computer system, the more the chance for a problem which will cause the entire computer system to fail. Therefore, there may be more delays and errors and the risk of a breakdown in the computer affects the entire organization rather than an unique function. Also, a consideration which is sometimes not adequately assessed is the effect of the computer centralization on managerial performance and motivation. When a manager has his own computers, he is, in fact, totally accountable for results. When he has to rely on a computer system that he does not control, he can lose the ability to make meaningful commitments. Today, the proponents of centralized computer systems are winning. There is a definite tendency to centralize and thereby reduce the higher costs of many smaller computer systems. As time progresses, this cost differential between one large system and many smaller systems will increase. It is also generally recognized that the limitations inherent in having one large computer system can be managed through experience and the continual improvement in appropriate communication channels.

A trend which is becoming much more common is the total systems approach. It is not new but has been receiving recognition in the commonly heard terms - "management information system", "integrated approach", "total information system", or even "on-line, real-time computer system". Basically the concept proposes that a system consists of many interdependent sub-systems which are interlocked together, perhaps with the use of a computer. This total system is an integration of all major operating systems within a company and incorporates the information needed for planning and control so that timely and accurate decisions can be made by managers of all levels. However, it is important to realize that everyone views the concept in a different manner, dependent upon their perspective.

One of the tenets, held by many, is that a computer is a necessary and vital ingredient to the total systems; however, is the computer a necessity in the total system concept? If one simply takes a look at General Motors or Ford in the 1920's, one will quickly realize that their organizations embodied the total system concept. However, it is important to point out that today's modern computer does enhance the value of the total system. It provides the means of implementation and then maintenance. Therefore, through evolution the computer has become an ingredient in the total system concept.

The planning and control function, much like the computer system, can vary in degrees between being centralized or decentralized. Looking at it from the perspective of a division manager, he is losing his control in direct proportion to the degree of centralization. Another argument for decentralization is that the planning and control group in each division has access to the day-to-day real world problems. It is thereby able to develop and implement the most realistic plans. However, there should always be some degree of centralization to provide overall procedures, policies, and programs. There are also better economics in having just one centralized function. This centralized function can afford to provide the specialized work which will attract the capable professional. There would be less duplication of effort and the organization would be planned as a balanced whole. During the past few decades, there has been a tendency to decentralize the planning and control functions to the various divisions. However, with the advent of large integrated data bases, there is an excellent opportunity for centralized planning and controlling, but with decentralized operations. Through the use of real-time, on-line systems, the operations manager in each division can interrogate the centralized data base with his own terminal. He can then see the results of alternative decisions and thereby make the best decision.

It is possible through the use of integrated data bases to mold or split almost any function in or across division boundaries. There are many imaginative ways in which this can be accomplished. It must always be remembered that man designs the computer systems to carry out the function he wants. Never does the computer system dictate that man must carry out prescribed functions in a prescribed manner. In this concept, management has the option to utilize the computers in any manner. They can determine the degree of centralization. Since it is equally possible for the operating personnel as well as the top management to use the same common data base, the need for middle management will decrease. This will tend to flatten the organization.

CHAPTER II

PROBLEM FORMULATION

Statement of Problem

Currently, most management information systems do not meet their predefined goals on schedule or within the planned resources. The problem has not been in the "technical aspects" of the system, but rather in the planning and control process of answering the basic questions of how, who, where, why, and when. The problems can occur in any one of the complex set of interrelated activities which constitute the management information system. However, the problem is usually intensified if it must be corrected in the testing activities. Historically, the problem has been viewed as being independent of all other activities. The emphasis has usually been placed on the activity in trouble, many times at the expense of the other activities.

Through gathering data on various management information system projects in International Business Machines, the approximate percentage of resources and elapsed time estimates over the eight phases assumed to constitute the eight phases of the MIS has been established and are portrayed in Figure 2. Chapter III develops the framework for the eight phases along with appropriate descriptions. It should be emphasized that the hump in the installation phase reflects the added resources required for cutting the new system over to production. No matter how well planned and tested the system is after system test, there is

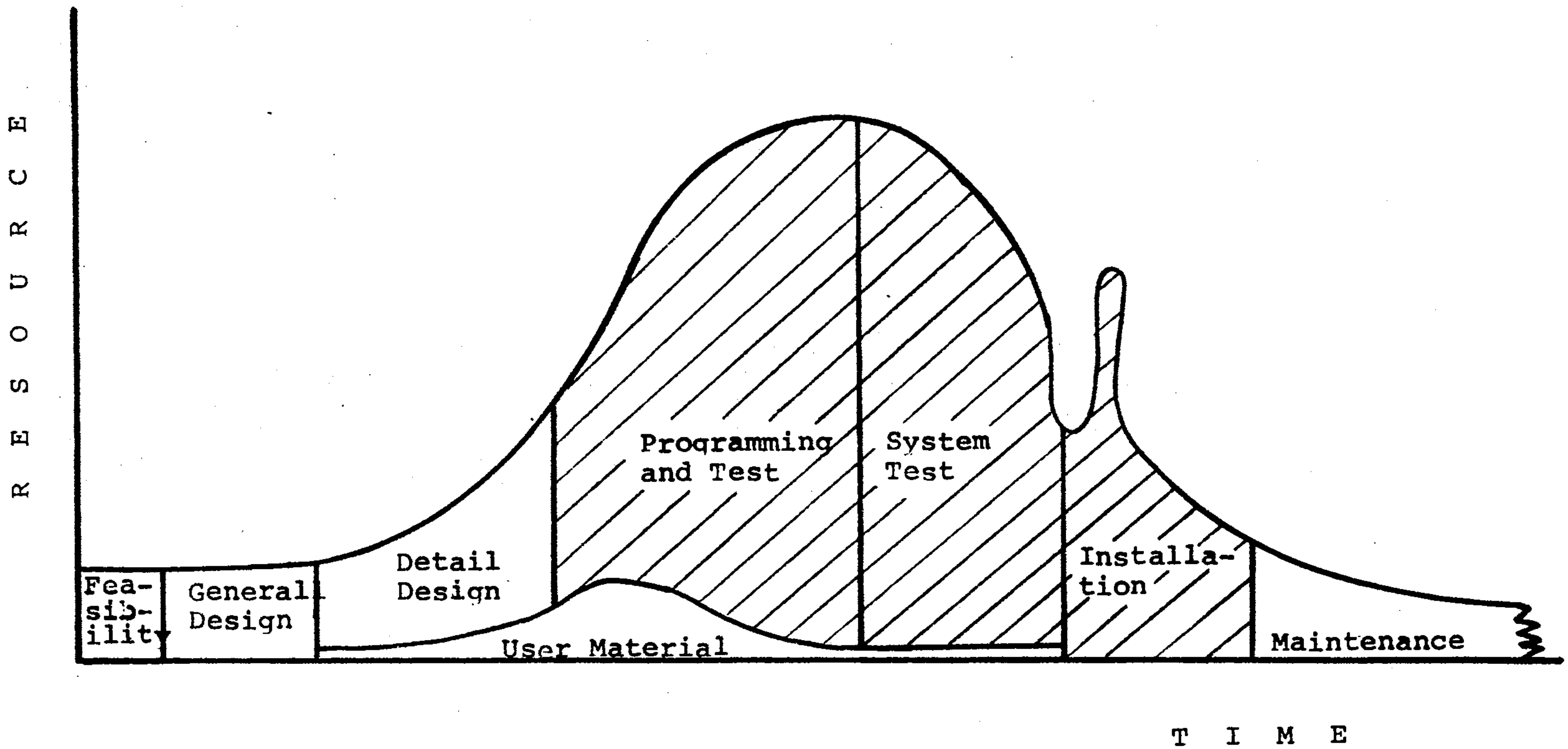


Figure 2. Resources and Phases Versus Elapsed Time

usually a need for additional resources. As can be seen in Figure 2, the cross hatched testing activities in the programming and test phase, system test phase, and installation phase, account for a very high percentage of elapsed time and also resources. Testing is defined to mean all the activities after the programs are coded through actual reliance (cut-over) at the installation location. Therefore, testing is not limited only to checking-out the programs to insure their correct functioning, but also to the factors which will either enhance or limit their successful installation. In the past, problems in the testing activities have been the most disasterous. A few of the potential problems are found in Figure 3.

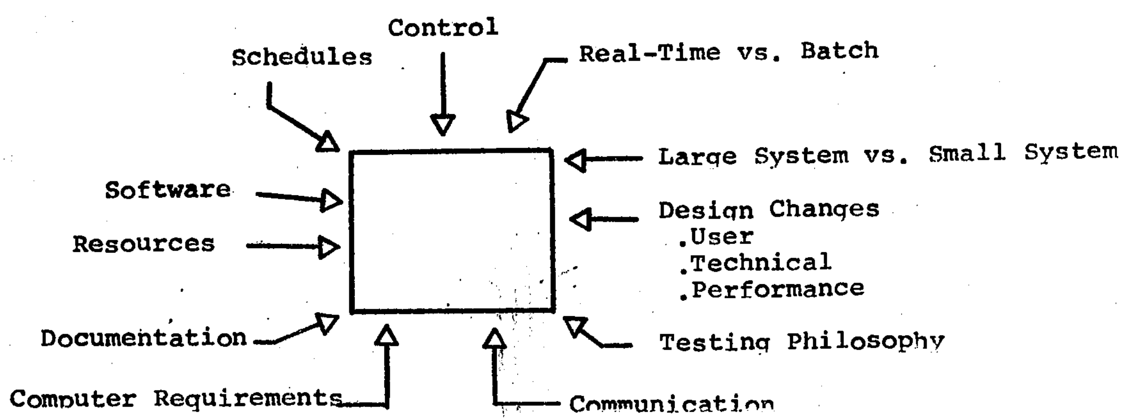


Figure 3. Factors Affecting Testing

Some of the factors affecting the testing process are described in this section at a general overview level. The detail analysis and discussion of all the factors is found in Appendix A which contains the activity descriptions for the general model.

The resource expenditures during the testing activities are extremely high. Not only are there personnel costs, but there is also considerable computer time required to insure that the programs, applications, subsystems, and system function according to specifications. The resource required follow a cumulative curve similar to the one in Figure 4.

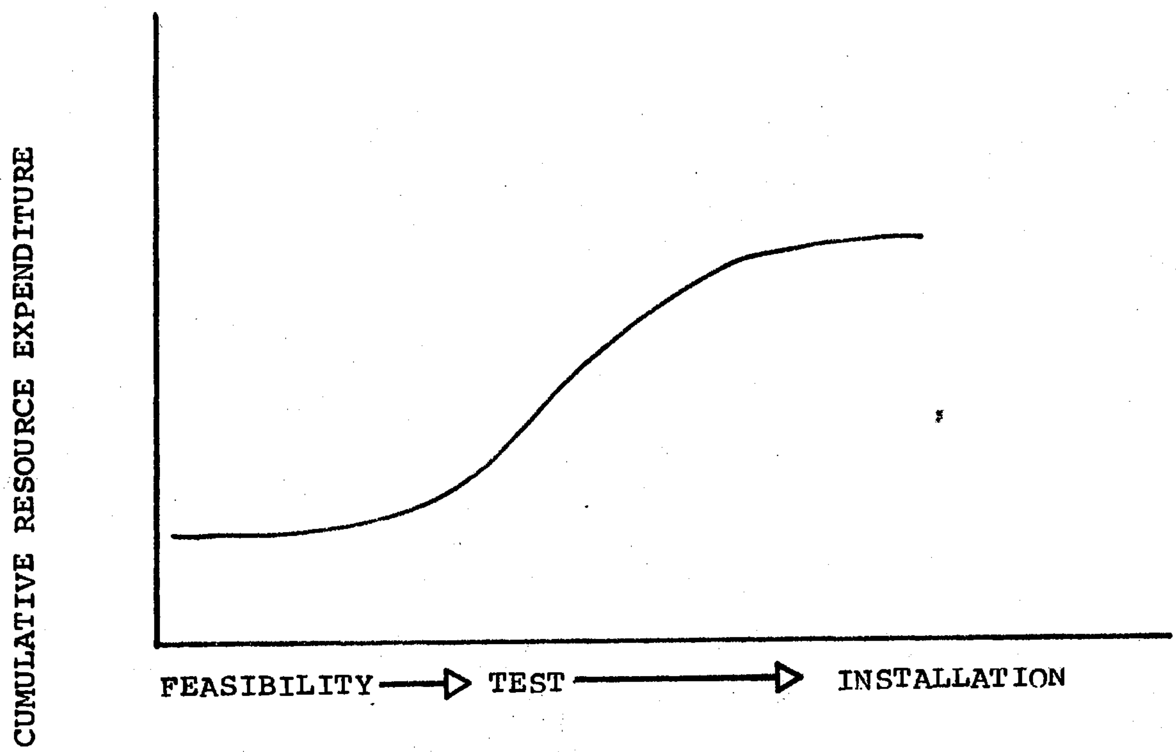


Figure 4. Cumulative Resource Expenditure Versus Phase

It is much easier to test a management information system which provides few straightforward functions than it is to test a large system which provides many complex functions. As the scope of the management information system increases, the complexity increases proportionally. This, in turn, forces the resources to increase at an exponential rate as depicted in Figure 5.

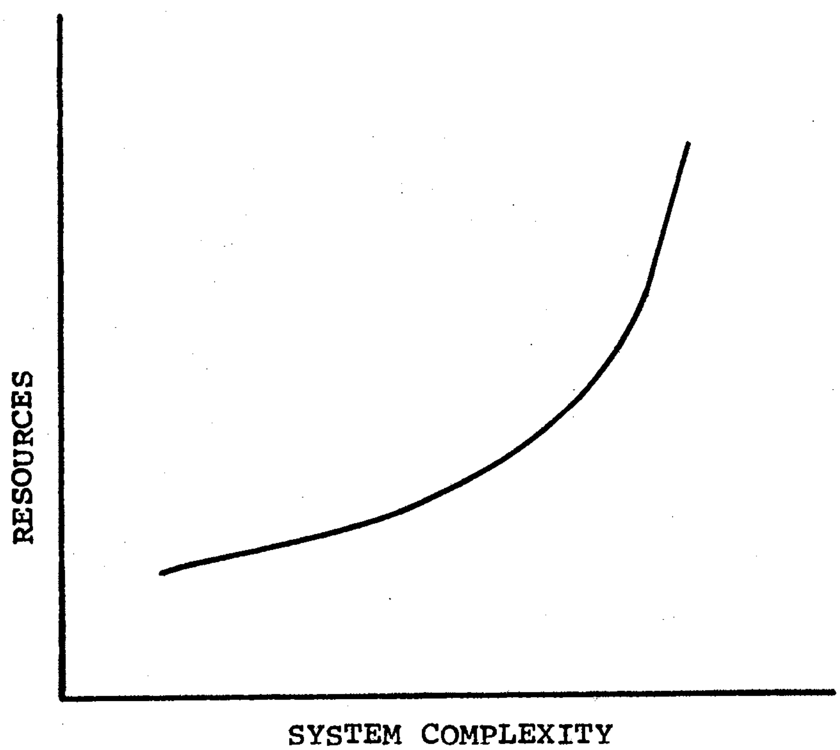


Figure 5. Resources Required Versus System Complexity

To more definitively define the size of a management information system, a comparison between small and large is provided in Table II.

It must be noted that the comparisons are of application source code not including comments.

TABLE II

COMPARISON OF SMALL VERSUS LARGE SYSTEM

Size	Approximate Number of Lines of Code
Small	300,000 or less
Medium	600,000
Large	900,000 or more

Another consideration required with large systems are documentation requirements. In a small system, documentation can be done by word-of-mouth or in small meetings. However, when there are larger groups (10 and up) involved, it is necessary to follow standard documentation guidelines. The testing phase should insure that the appropriate documentation is not only written, but that the documentation corresponds to the system's performance and to the documentation standards.

The nature of "real time" adds an additional concern in the testing phases. The response time of a real-time system is usually measured in short time intervals - seconds or parts of seconds. With a batch system, the response time is usually measured in longer time intervals - hours or days. The testing problems become more difficult to find with real-time in that the computer may become inoperable due to an infinite number of transaction combinations. Another factor is that real-time generally implies on-line and time-sharing capabilities which implies a large centralized computer. When a large system becomes inoperable, the problem is difficult to define and the solution development a complex process. The batch system being generally smaller and less complex is easier to fix.

Usually, it is not possible to develop and install a large, real-time system all at once due to schedule and resource constraints. The normal approach is to develop the system in predefined releases which can be made operable at the installation locations in installation steps. As can be imagined, this causes a complexity of interrelationships between programs, applications, subsystems, releases, and logical installation units. Synchronization is quite complex when attempting to implement a meaningful schedule which will satisfy installation requirements. Although eventually the entire management information system, which is composed of all the releases, will progress through all eight phases, at any one point in time the programs can be in different phases. The programs make up an application which is considered to be a manageable, trackable, piece of code with a defined function. A subsystem consists of a group of similar applications. Various applications are brought together during the testing phases to form releases which are provided to the installing locations in terms of installation steps. The building of a management information system is depicted in Figure 6.

Change requests are an ever present workload generator which must be considered. Every effort should be made to understand this activity

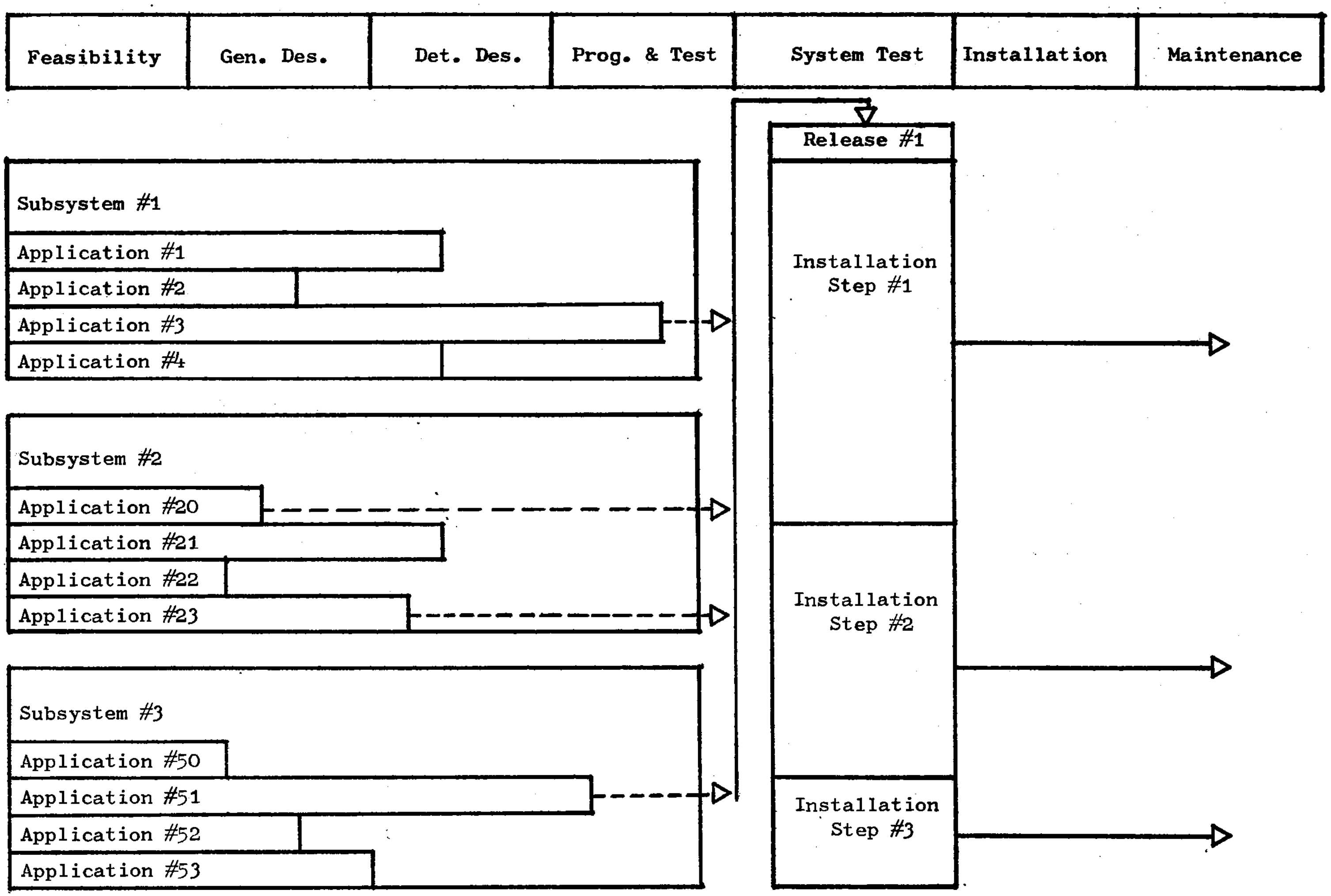


Figure 6. Building a Management Information System

because it will affect the quality and duration of the testing. Requests for changing the design may come from any one of several sources. The request may be based on providing design enhancements, performance improvements, technical compatibility, etc. Generally, the risk of satisfying the users with regard to design enhancements is dependent upon how far the system has progressed. This is shown in Figure 7.

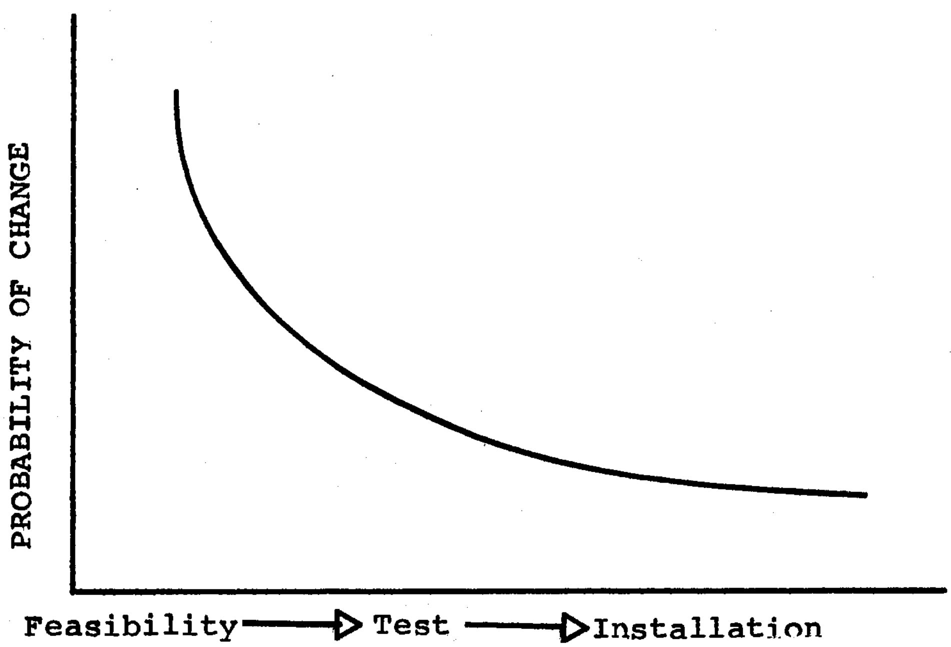


Figure 7. Risk of User Dissatisfaction

The risk of making design changes and performance improvements because of technical problems also decreases over the system's progress, but follows a different pattern. This trend is depicted in Figure 8.

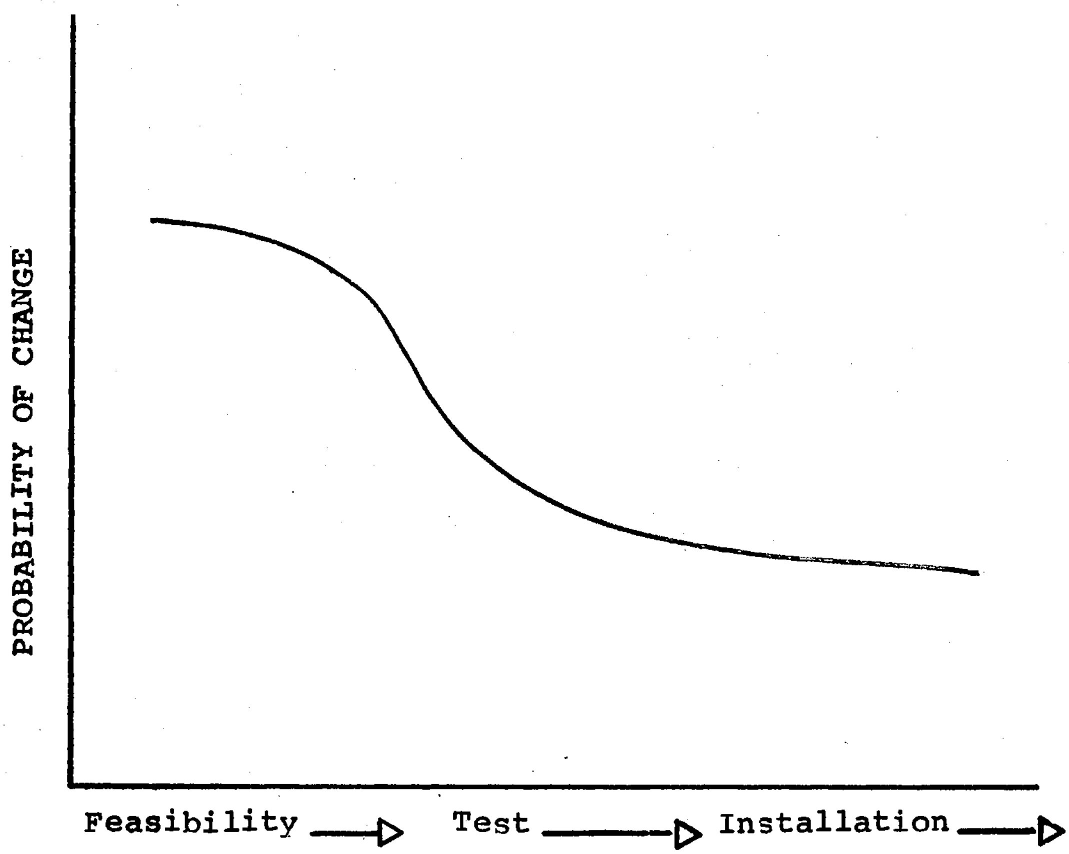


Figure 8. Risk of Technical Problems

Probably the most critical problem in insuring a successful system is the development of a comprehensive testing strategy and then the communication of that strategy to all affected areas. Barnard (2) points out that "An organization comes into being when (1) there are persons able to communicate with each other (2) who are willing to contribute action (3) to accomplish a common purpose." The common purpose assumed to bind men together in this dissertation is the installation of a successful management information with schedule and resource constraints. One of the essential ingredients is a comprehensive testing strategy. There are many different, complex technical terms which are applied to various testing activities. But, if one goes beyond these technical terms, a similarity among the various detailed activities between various systems becomes evident. Communication of the precise meaning and makeup to each of the technical terms is, in fact, one of the major problems.

Purpose of Study

There is a critical need to understand large, real-time management information systems so that its resources and schedules can be successfully planned and controlled, especially in the complex, expensive testing activities. The purpose of this dissertation is to develop a general management information system model and then apply it to a specific situation so that the stated critical need can be accomplished. It should be understood that the purpose is not to provide an immediate answer to a specific problem, but rather to assist in the development and analysis of alternative solutions through the application of a computer-based model. The goal is to provide management a technique for understanding and then managing the increasingly complex activities in the eight system phases - emphasizing the testing activities.

It is not possible to develop a generalized model which is directly applicable to solve all problems in all management information systems. However, through minor modifications to the generalized model, all management information systems can be appropriately modeled so that alternative decisions can be evaluated. This is primarily because there is a similarity in generic activities from one management information system to another.

Literature Search

There is a considerable amount of literature concerning the various phases inherent in management information systems. Blumenthal (3) describes management information systems from a general overview level while emphasizing that there is a framework through which each management information system passes. One of the best descriptions of the various phases is done by Benjamin (4). He breaks the development cycle into various smaller tasks and then defines and discusses each one. However, neither Blumenthal or Benjamin discuss the phases involving testing activities to a meaningful level or discuss the phase relationships in a modeling context.

There is also a considerable amount of literature concerning the management aspects of management information systems. Szweda (5) defines the role of the data processing manager in the planning, organizing, evaluating, and decision making processes. He discusses the delegation of authority, line and staff functions, personnel selection, etc. in great detail. Another excellent source of the management aspects, especially with regard to historical development of management information systems, is Sanders (6). Another valuable book which discusses how to organize, staff, and manage the management information system is by Albrecht (7). He also provides a case study on the development and implementation of an organization for a large company.

Most of the literature has been in reference to batch-mode management information systems. There is considerably less literature concerning real-time and even less concerning large, real-time management information systems. Although some of the principles concerning batch-mode systems apply to real-time systems, the real-time system is really an unique entity which has its own characteristics and considerations.

There has been a considerable amount of published and unpublished literature concerning the interactions between the software and hardware of a management information system. Martin (8) (9) and Flores (10) (11) are probably the most widely known. They have discussed the various technical aspects of computer programming as well as the physical design of the computer systems. To gain an understanding of their ideas and concepts is mandatory before one can attempt to plan and control a management information system.

Currently, there is a lack of literature concerning the development and application of a computer-based management information model which considers all the activities in an interrelated framework.

Procedure

There are many procedures which can be used to develop this dissertation. Each procedure has its own advantages and disadvantages. The procedure selected is depicted in Figure 9 and is described by discussing each step.

Step #1 includes all the activities which are necessary before the general model can be constructed. This includes a thorough search of the literature, statement of the problem, definition of the dissertation purpose, and selection of the most appropriate computer modeling

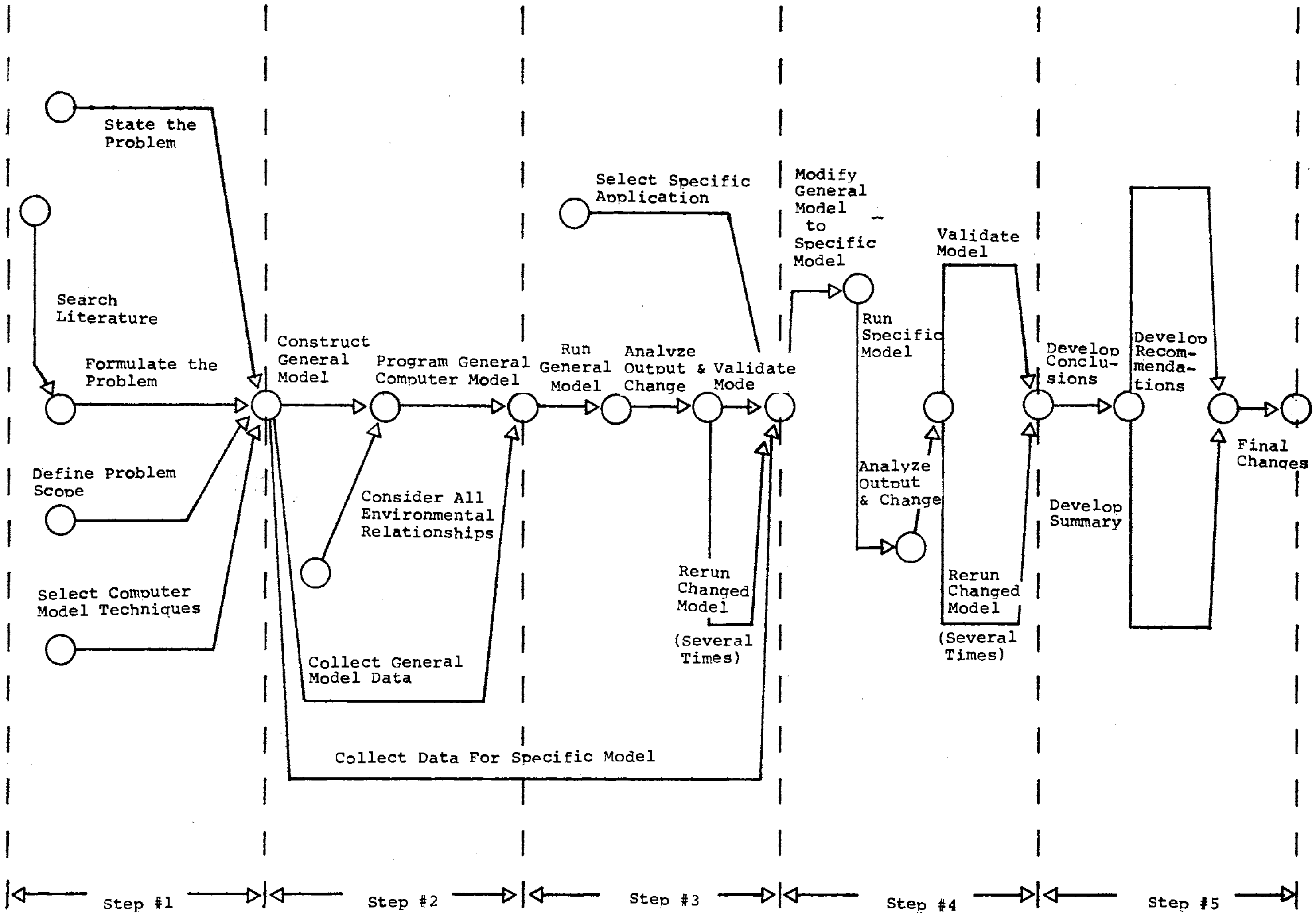


Figure 9. Dissertation Methodology

language. The material developed in Step #1 is provided in Chapters I, II, and III.

Step #2 consists of the activities required to develop the general model before it is run for the first time. This step equates to the activities required to design and code a management information system. It includes the construction of the first-pass general design network model, the collection of the elapsed time and resource data for each activity, the consideration and statement of all environmental factors, and the coding of the general model. The material developed in this step is found in Chapter IV and also in Appendix A.

Step #3 consists of the final development and the preliminary validation of the general model. The general model, which had been designed and coded in the previous step, is loaded on the computer via an IBM 2741 terminal. Then, the model is debugged with regard to input errors. After the input is debugged, output reports are generated. The output is then analyzed for model logic errors. After insuring that the logic errors are corrected, the output reports are analyzed with regard to the effects of the data elapsed time durations. If there are problems, the data has to be analyzed and evaluated to insure that it is compatible with the model logic. The quickest way to determine data validity, in this case, is to evaluate the activities on and near the critical path. If the activities on the critical path are in fact the critical ones, the logic and data are correct. The material developed in this step is found in Chapter IV and also was the basis for the first-pass of Appendix B.

Step #4 is where the dissertation's previous work is applied to an actual management information system. In the application, it is necessary to alter the previously developed general model to reflect the real-world situation. The application of the general model to a specific management information system causes a new specific model to be developed. The specific model, by design, must possess the general model considerations. As with the general model, the specific model must be designed, coded, tested, and validated. The validation process of the specific model is easier to accomplish because there are definitive schedules and end products for each of the activities. The validation process of the specific model in effect also validates the general model. The material developed in this step, including the description of the specific management information system, is found in Chapter V.

Step #5 is the last step and it includes the development of the summary, conclusions, and recommendations. The summary provides a brief composite of the entire dissertation, the conclusions are the results of the dissertation, and the recommendations are those items which should be investigated further. This material is found in Chapter VI and provides an overview of the dissertation.

CHAPTER III

MANAGEMENT INFORMATION SYSTEMS

Definition

The term "Management Information Systems" connotes various meanings to different people. In an effort to avoid any problems in understanding its composition, various definitions will be provided. Then, a definition is developed which will be used throughout the remainder of this dissertation.

The following management information system definitions are provided:

> In summary then, a Management Information System develops to the needs of management for accurate, timely, and meaningful data in order to plan, analyze, and control the organization's activities and thereby optimize its survival and growth. The MIS accomplishes this mission for management by providing means for input, processing, and output of data plus a feedback-decision network that helps management respond to current and future changes in the internal and external environment of the organization (12).

> It should be clear at this point that the management information system is the catalyst and the nerve center of the organization. Moreover, it is the common system that permits the four other resource systems (money, manpower, materials, machine and facilities) to function as an integrated whole. It performs this integrative role in four ways: (1) it provides information among all four systems regarding the impact of each on the whole; (2) it establishes sensors and control measures for data acquisition required by the other four systems; (3) it maintains central data banks relative to the decision processes of the other four systems; (4) it generates output information on demand and on an exception basis that reflects the operation of all systems, including the management information system (13).

The effective MIS does specific things in response to specific requirements. It furnishes relevant data in useful form to the right person, at the right time, for use in management decisions. MIS is the system which generates that information often already supposed to be in the hands of management (14).

A MIS is a system that aids management in making, carrying out, and controlling decisions (15).

As one can see, there are many different concepts of a management information system. It can range from an extremely simple to a large, complex, computer-based system. However, it should be noted that there are also some similarities. Almost all definitions either imply or state that there is some type of input data, processing data, and then output information. This is visualized in Figure 10.

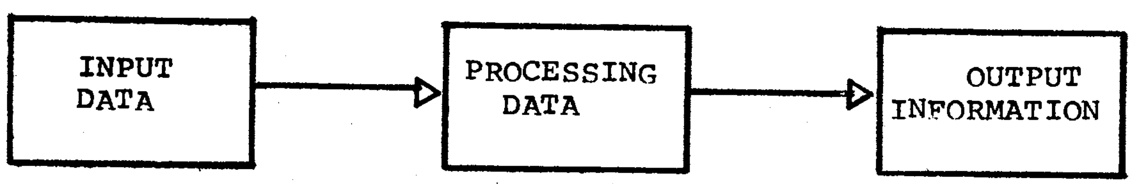


Figure 10. Elementary Management Information System Components

Therefore, if a person is "thinking", there is, in fact, a management information system. The person has received some type of stimulus (data input) which has caused him to think (processing data), and he will either take immediate action or wait for more stimuli before taking action (information output). This is a rather simplistic system. Larger systems could include a person reading a clock, one person talking to another, a group of people discussing politics, engineers tracking the progress of a spacecraft, etc.

The management information system definition used in this dissertation assumes that input data, processing data, and output information are necessary ingredients but also that a real-time, on-line computer system (implies common integrated data base) is utilized in an environment to affect the decision making process of one or more people. On-line means that the point-of-origin devices can communicate with the main processor and the main processor with the various remote devices (16). Real-time is a relative concept but if the time between the entering of information into the computer system and the logical response to this input is insignificant, then the system can be said to be real-time (16). The word insignificant may be measured in terms of seconds, minutes, or hours, depending on the unique application. The key is that the output information affects the decision making process. Although the definition developed in this dissertation is the opposite extreme from the simplistic management information system, it is where the greatest benefits are forthcoming. It is here where the state-of-the-art techniques are being developed to assist man in almost all facets of the environment.

Framework

Most management information systems progress through various overlapping phases. If one compares these phases between different management information systems, a similarity will evolve. This commonality starts with the initial feasibility discussions and ends, assuming that it progresses through all other phases, when the management information system has been installed and is being maintained. In most situations, these phases do not have clearly identifiable start and stop points. However, if enough detailed planning is done, it is possible to break a management information system into distinct serial phases. This is depicted in Figure 11.

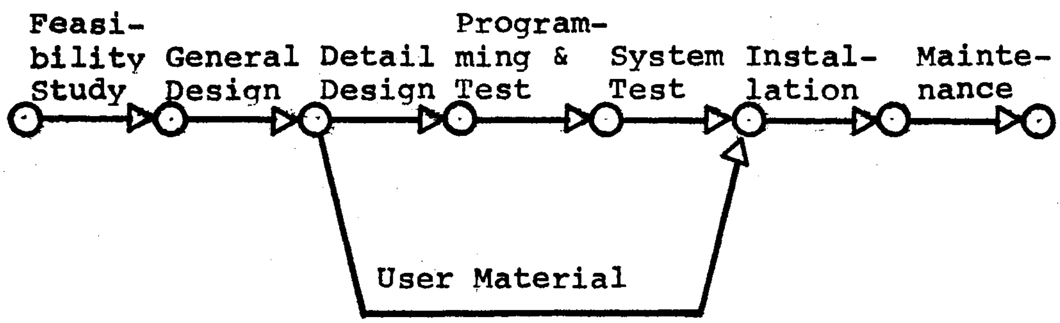


Figure 11. Management Information System Phases

Although eventually the entire management information system will progress through all eight phases, at any one point in time its various subsystems (meaningful combination of user functions) will be in various phases. By breaking the MIS into subsystems, it is possible to structure manageable entities which can be assigned, designed, tested, and installed.

The planning and control of the complex subsystem interrelationship becomes extremely critical to the overall MIS success. As one can imagine, within each phase there is a myriad of activities which must be considered. The synchronization of each of the potentially thousands of activities in developing a meaningful schedule from the initial feasibility study through the maintenance activities is a demanding challenge.

There are generic activities which must be accomplished in each of the MIS's eight phases. To understand the overall MIS framework, a figure depicting a particular phase followed by a general discussion of the generic activities in that phase is provided.

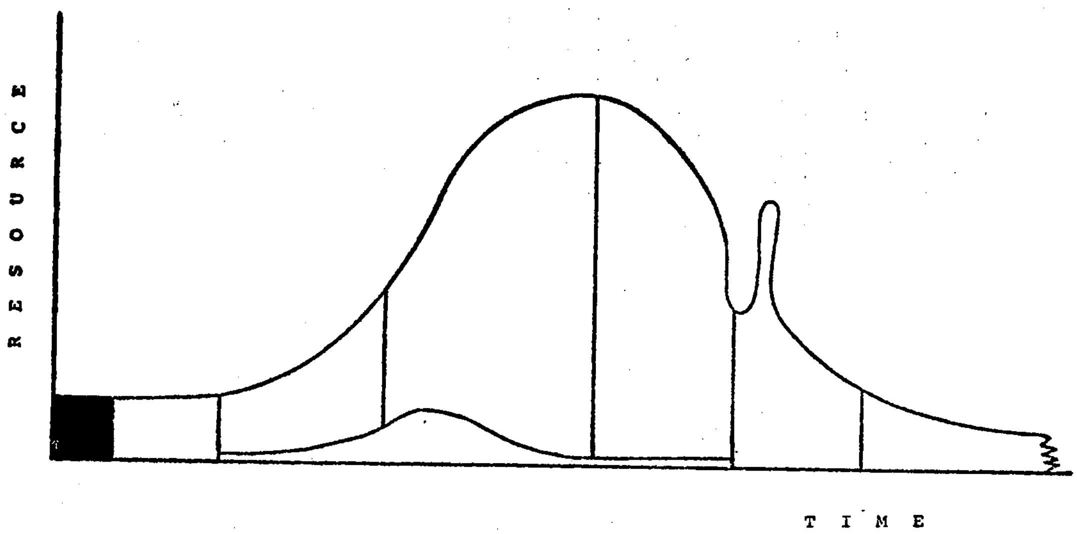


Figure 12. Feasibility Study Phase

Feasibility Study

The feasibility study just does not happen - it requires knowledgeable personnel in the area being investigated and the backing of top management. The first step should be a preliminary feasibility study. This is a study which would take place before the full feasibility study and would accomplish the following: develop the reasons and objectives for the feasibility study, estimate costs of making a full feasibility study, select appropriate personnel for feasibility study, and develop a feasibility study schedule. For the preliminary feasibility study to be worthwhile, the general goals and objectives of the overall data processing program must have been previously defined so that the proposed project can be analyzed for compatibility. If top management supports the preliminary feasibility study, the full study begins. The first item to investigate is the present system and its recurring costs. This can be accomplished by interviews and examination of records. It must include the present procedures and determination of transaction volumes. Then the proposed system must be evaluated with regard to how it would operate. Also, if there are any other proposed systems which should be investigated during this analysis, they must be so stated.

In all cases, the operational, technical, and economic aspects must be considered. The operational considerations include whether the new system could in actuality produce the required output within the response time requirements. The technical considerations analyze whether or not the necessary software, hardware, and other support requirements are available or will be available when required.

The economic considerations are of great concern. Included in the economic analysis must be dollar values placed on computer time, travel, relocation, manpower, contingencies, burden or overhead consideration, and other division or corporate support. All assumptions must be stated so that it is possible to know the base upon which the estimates are based. The assumptions will be needed later to cross-check the planning assumptions. Any benefits which are derived from the new system must be stated as being either tangible or intangible. For each benefit, a statement of cost and savings must be identified. In some situations, the intangible benefits are of the most importance and, therefore, the decision will be to develop and install the system. Also, because of technical considerations, it may be mandatory to develop the system. If the proposed system does not improve the organization's profit picture by having a good return on investment (ROI) and a short-term payback period, it should seriously question the MIS project. The results of the feasibility study are presented to top management who will make the decision on whether to continue the project.

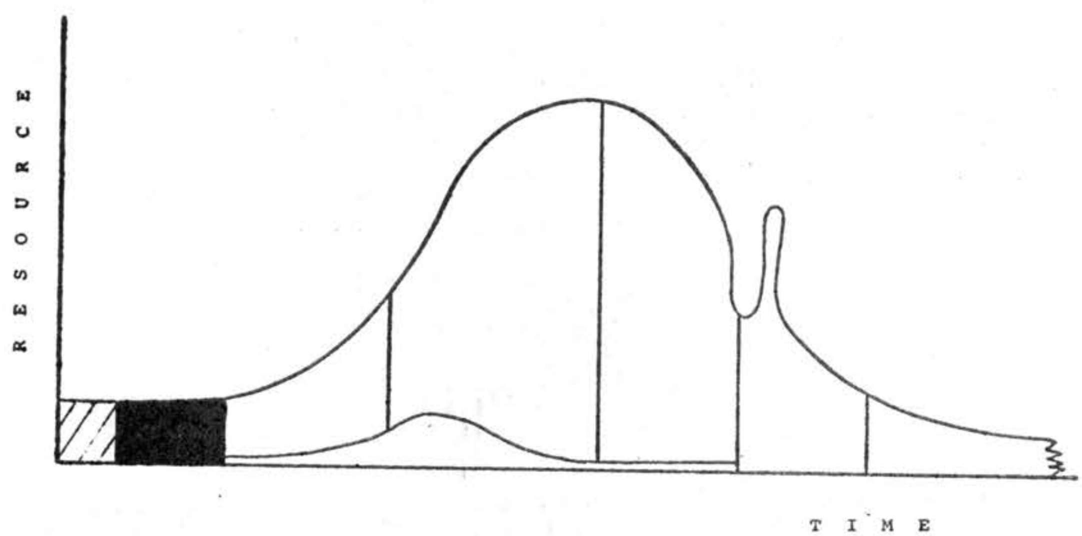


Figure 13. General Design Phase

General Design

This phase consists of a very high level design which identifies the scope or functional boundaries, significant features which trace the function from initial initiation to completion, development strategy, installation strategy, and user services. It also includes a description of the philosophy and basic concepts of the system and its various subsystems, a list of the committed and approved functions which constitute the subsystem; a list of user requirements which, although not presently committed to, are recognized as desirable, and which will be considered in later stages of development; and a list of user requirements which are not being satisfied within the presently committed design of the subsystem, and which are not intended to be considered in the future according to present plans. Statement of the extent of user control of input and what the user must supply to achieve the desired output, the general processing rules (commonly called regulators) for which the user supplies control information, and the degree of customization possible. The fallback and recovery philosophy must be indicated if the system for some reason fails to operate properly.

A flowchart should be developed which defines the normal functions which are performed. It should include the man/system, system/system, and system/man interfaces.

In all cases, the general design should include necessary exhibits, diagrams, and flowcharts to make the material meaningful.

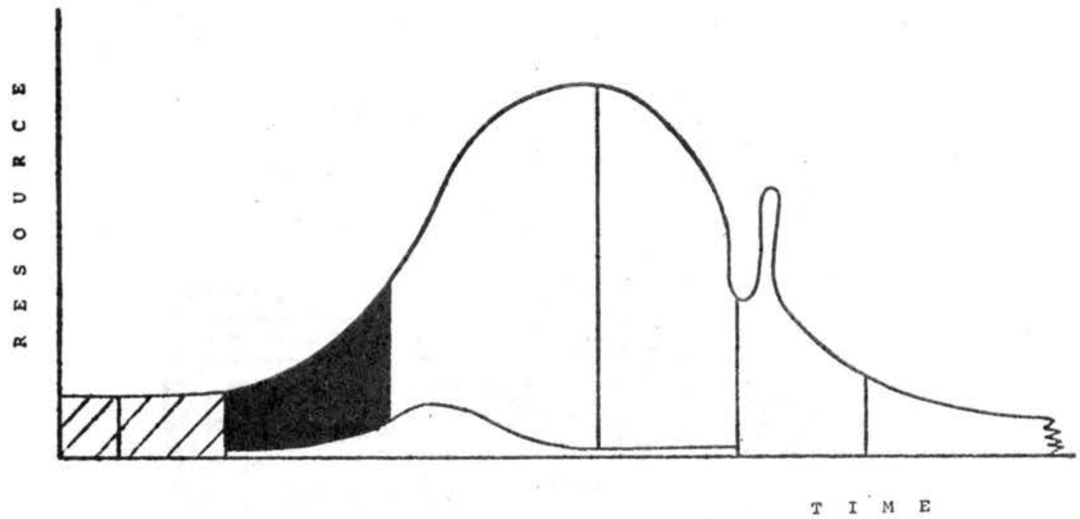


Figure 14. Detail Design Phase

Detail Design

This phase consists of a detailed description of the sequence of events which describe what the system does and also the necessary data processing steps. Where the general description discussed general functions which take place, this phase describes the applications (groups of one or more programs) which make up the function. It includes a description of the entities that cause the application to start processing (initiators), a list of the records that are updated, deleted, or referenced by the application, and the output which could be a message, report, or a transaction (secondary transaction) to another application. A logic flowchart should be developed that will show the sequence of programming, the decisions and actions that take place during the processing, and the sequence in which the data is used. Accompanying the logic flowchart should be a narrative which describes the logic being depicted, why the logic is necessary, and the appropriate data sources.

In addition, all the programs which make up the individual applications must be listed and adequately described. The program number and name, function, purpose, and structure must be stated. Then, as for the application, the program's input, records, and outputs must be described. Next, a program flowchart and flowchart narrative must be developed. As with the general design, appropriate exhibits, diagrams, and flowcharts must be developed according to predefined standards.

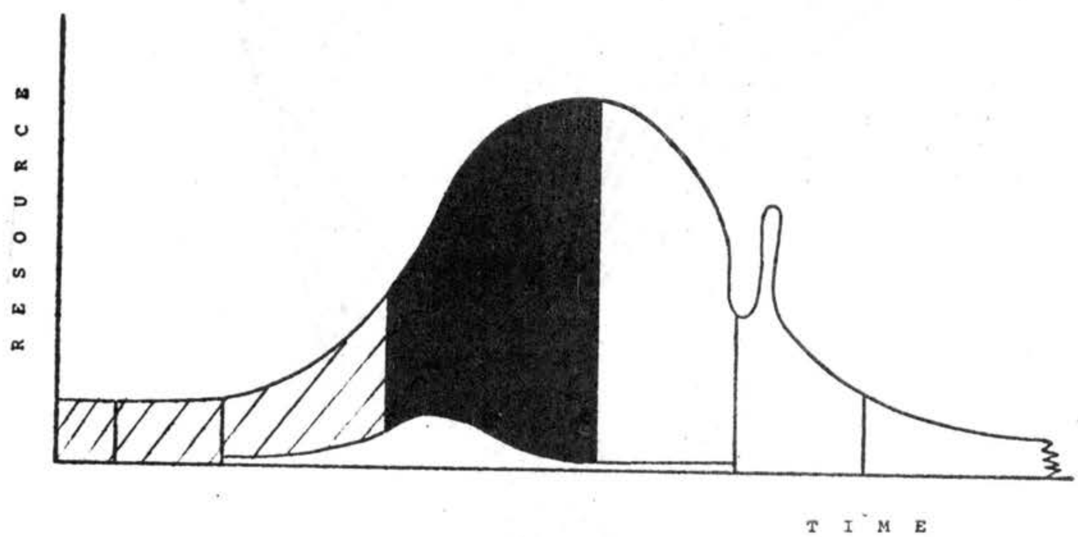


Figure 15. Programming and Test Phase

Programming and Test

The logic flowcharts are expanded into a set of instructions (coding) in a defined language. Then the coding is keypunched and verified. Next, there should be a desk check which looks for illegal expressions, erroneous data references, program logic errors, program inefficiencies, and other deviations from specifications. Then, the program is assembled or compiled to check for coding errors. The errors must be corrected until the first error-free compilation is obtained. The program must then be tested against a predefined test plan until satisfactory results are obtained. The tests will be increasing in size and function as the program successfully passes the various test plans. The first computer-based test will probably be a unit test where the program is tested by itself. Then a functional test is done to insure that the programs within a given application and eventually a subsystem meet the test plan criteria. Then an interface test is done which tests the compatibility of the various subsystems.

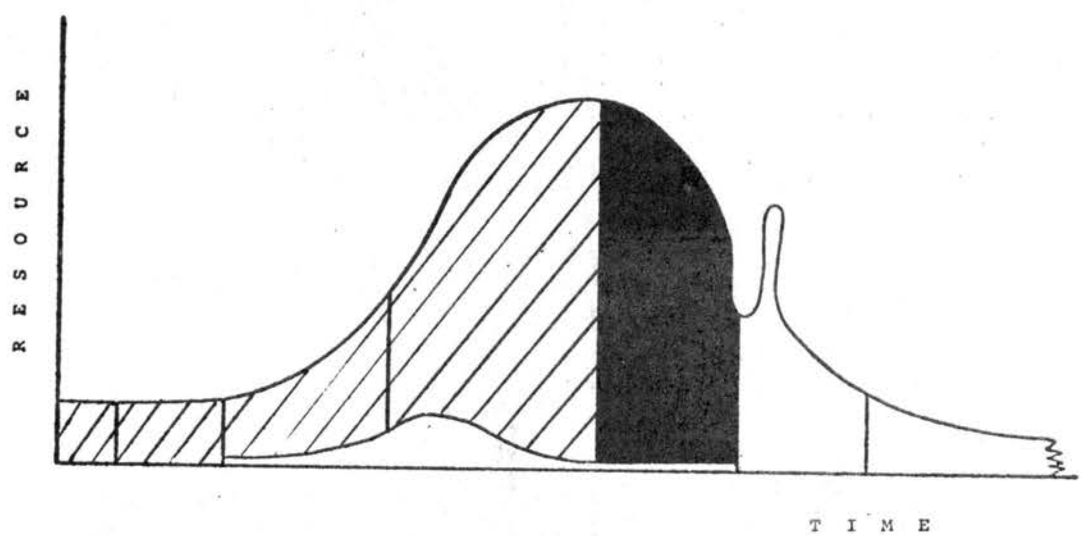


Figure 16. System Test Phase

System Test

This phase is extremely critical to the success of the program. It includes all the work necessary to test the systems, subsystems, applications, and programs under realistic operating conditions. Appropriate test plans with predetermined results should be developed and used to insure that the programs perform according to plan. When problems are noted, they must be corrected and the test case rerun until satisfactory results are obtained. Then, the documentation must be updated.

Another activity which should be undertaken during this time frame is a simulation of the installing locations requirements. It would include appropriate transaction volumes, timing characteristics, data bank record levels, etc. From this, design problems may be encountered, performance characteristics can be measured, and other criteria can be evaluated.

There are many different types of tests which can be utilized to determine whether the system meets specified criteria. However, there must be a testing strategy and philosophy which everyone is working toward and each type of testing should lead to the overall objective.

User Material

There are various types of support which users should be provided to assist in their understanding and operation of the system. This includes user procedures, reports and inquiries documentation, operations procedures, education requirements, and dictionaries. It is important that the users review all their documentation to insure that they can understand it. Therefore, the language and terminology of the user must be used in all situations.

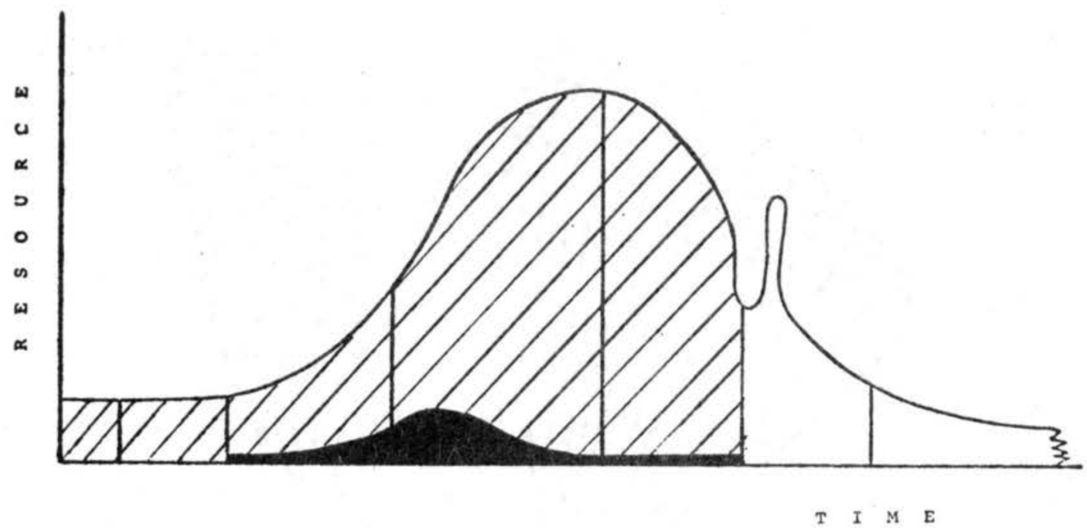


Figure 17. User Material Phase

The user procedures should describe how the users carry out their function. The first item is to define the user's job description by stating the job name and job purpose. Then, the transaction's (input from terminal) name and purpose are described. If there are any items which must be accomplished before the transaction can be executed, it must be so stated. Any interfaces that the transaction may have with other transactions that may affect its processing must be listed. Then, the hardware (terminals or other input devices), which executes the transaction, must be specified by name and type. The mode of operation (updating, adding, or deleting) must also be stated. The procedures for entering the transaction must be defined. Finally, a list of all output resulting from the transaction must be identified. The output includes error messages, action-required statements, etc.

Also, there must be appropriate report and inquiry documentation. It will assist the operating personnel in understanding the system. The description of each report and inquiry should include its purpose, frequency, security requirements, restrictions, interfaces, entry procedure, hardware requirements, and output messages.

The operation instructions must be provided to describe how to operate the system under normal circumstances, how to recover and restart the system after problems, and how to answer questions when portions of the system are shut down.

There also must be a dictionary which will define various aspects of the system. For instance, there should be a glossary which contains acronyms, abbreviations, and definition of terms; a data element dictionary; a user record dictionary; a catalog on utility availability; and finally operator console messages. Before the system can be released to the installing location, the users must be educated in its use. This can be done through presentations and also appropriate documentation. The exact combination will be dependent on the audience. For example, it would be possible to structure a series of presentations which would begin with providing the total picture of each subsystem, the requirements for file initialization, and a statement of differences between the current system and the new system. The next level of education would include the complexity of the system, the number of bridges which must be built, the impact on the existing environment, the flow of information through the subsystem, an understanding of the key transactions, data elements, etc. The last level of education should insure that the users understand the information flow in each program.

The schedule of classes, number of personnel, type of personnel, amount of effort, etc., all must be identified and made visible to all concerned. This will avoid potential negative user comments on not being adequately prepared to use the new system.

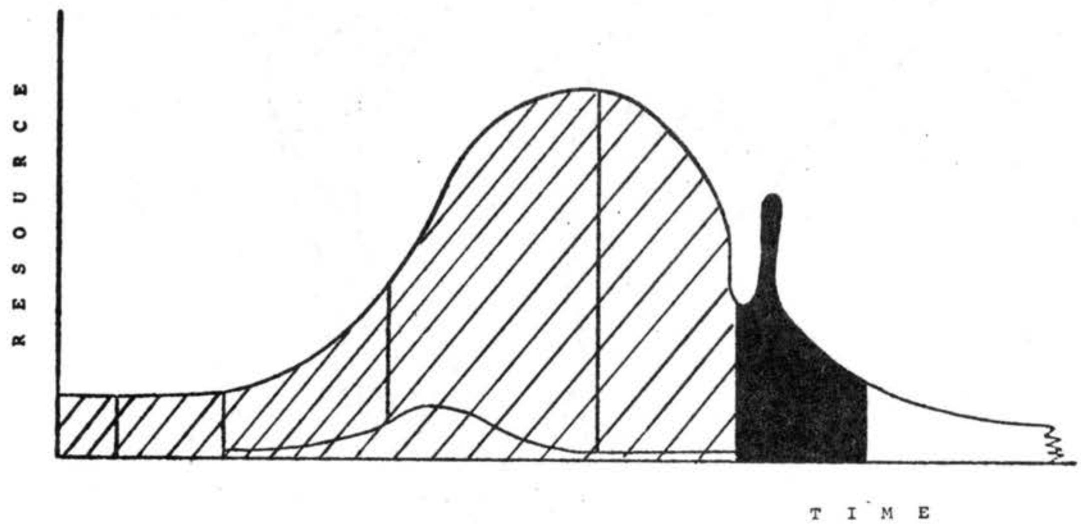


Figure 18. Installation Phase

Installation

Each installation location has its own set of unique problems which must be addressed. However, there is also common installation concerns among all locations. For instance, the functions to be provided in the new system must be understood, the bridges between the new system and the installation location's procedures must be defined. User concurrence tests must be developed and run. Support requirements for the installation activities must be sized and implemented. Parallel running of the new system to the existing system and also appropriate cut-over criteria must be established. Education and training must be completed, detailed conversion schedules must be developed and used to insure satisfactory progress, site preparation must be implemented, and appropriate hardware and software must be available. Cut-over (using new system) must be accomplished after satisfactory testing, parallel-run criteria have been met, and other planned activities have been completed. The hump during the installation phase reflects the resource expenditure required for cut-over. Whenever a management information system is cut-over to actually drive the installation location, there are numerous problems which were not known before, regardless of how much testing has been performed. The support will usually have to come from the location(s) who have developed the programs.

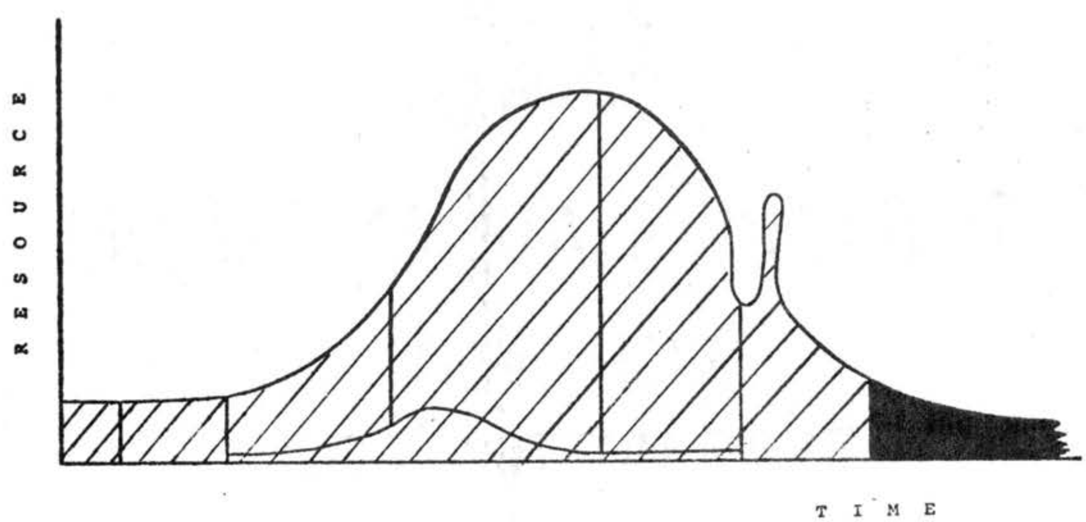


Figure 19. Maintenance Phase

Maintenance

One of the most important activities, which is sometimes not given enough attention, is the management information system maintenance which takes place after installation. A system just does not continually operate in a static environment. The environment is continually changing. Software is upgraded to enhanced levels, procedures are changed, performance improvements are desired, etc. All these changes must be considered and the system maintained appropriately. There should be a prescribed manner to handle each requested change to the system. Otherwise, havoc would result. For instance, there should be a formal request for change, statement of need, cost analysis, and a proposed schedule which is reviewed by appropriate management. There also must be a definitive time at which maintenance support stops. It must be agreed upon by the developer and installer.

Real Time Versus Batch

There is generally little confusion over the denotation of the terms "real-time" and "batch". However, there are commonly problems in understanding the connotation of the terms. Generally, a real-time system not only provides information to affect the decision-making process, but also implies being on-line and time sharing capabilities. An on-line system is one where an input device and output device are tied directly to the central processing unit. Thereby, data is sent directly from and back to the user location. Time-sharing implies that there is more than one user of the central processing unit at the same time.

In real-time operation, each transaction (data input) updates the integrated data base. Batch processing, on the other hand, holds all the transactions in a batch and then updates the data base all at once. The batch system has a response time which is usually measured in hours or days. The batch workloads are predictable and can be scheduled at specific times, whereas the transactions in a real-time system can be sporadic, instantaneous, and simultaneously entered from several terminals. In real-time the processing is done transaction by transaction. This usually requires the writing of special programs to handle the special contingencies. Also, programs must be written which will temporarily store and then return information in an accessible manner to the central processing unit.

One of the major advantages to a real-time system is that the user commitment is almost total. The user communicates directly through his terminal to the computer. The data processing department is no longer a middle man. With this direct user/computer interaction, the system will be providing the user with information within short time intervals so that it can assist in the decision-making process. Therefore, it is responsive to the user's needs. One of the most widely publicized and also one of the most successful applications of a real-time system was the IBM PARS system developed for airlines. With this real-time system, it is possible to determine customer information requests within seconds. It improved customer service, the percentage of seats which were occupied, and improved the performance levels in the sales and control offices. There are almost unlimited applications of real-time systems. For instance, it is now possible to control space flights to and from the moon, to schedule railroad freight yard activities, etc.

Organization

There are as many different ways to organize a management information system as there are people who express their ideas on the subject. There is not one ultimate way in which every management information system organization should be structured. Each situation is different and should be considered as being unique. However, there is an organizational approach which seems to be suited to almost all management information systems. There is also a set of generic functions which must be accomplished for the management information system to be successful.

Management information systems possess all the characteristics which lend themselves to the program management concept of organization. It is generally extremely complex, has continually changing technologies, contains tight time schedules, has resource limitations, is nonrepetitive, is a one-time effort, and is in a dynamic environment. Since the program management concept was developed to cope with these characteristics, it is recommended for use in the development, test, and installation activities. There is also a similarity in the necessary functions which must be considered. These functions are stated in Table III.

The program management organization concept would have the functions listed in Table III reporting directly to the program manager. This results in an organization which is rather flat and has many planned checks and balances. Its structure is depicted in Figure 20. The dotted line shows that the installation location generally does not report directly to the program manager.

TABLE III

MANAGEMENT INFORMATION SYSTEM FUNCTIONS

Application Development	- The design and test of the application programs.
Software Development	- The design and test of software programs (operating systems).
Test	- The test of the application and software programs before their being sent to the installing location.
Release and Control	- The receiving of the programs from the developing locations and the sending of the programs to the installing location.
Installation	- The installation of the programs at the installing location (usually a dotted line - does not report directly).
Design Assurance	- The insurance that the programs meet the system objectives, performance standards, and system specifications.
Documentation	- The review, publication, and distribution to support the programs.
Plans and Controls	- The establishment and monitoring of budgets, schedules, resources, goals, objectives, cost/benefit studies, etc.

Barnard (2) states:

> The system of communication, or its maintenance, is a primary or essential continuing problem of a formal organization. Every other practical question of effectiveness or efficiency - that is, of the factors of survival - depends on it.

By utilizing the project management organization concept, the communication channels are on not only a vertical basis, but also the horizontal. There must be a great deal of horizontal communication because

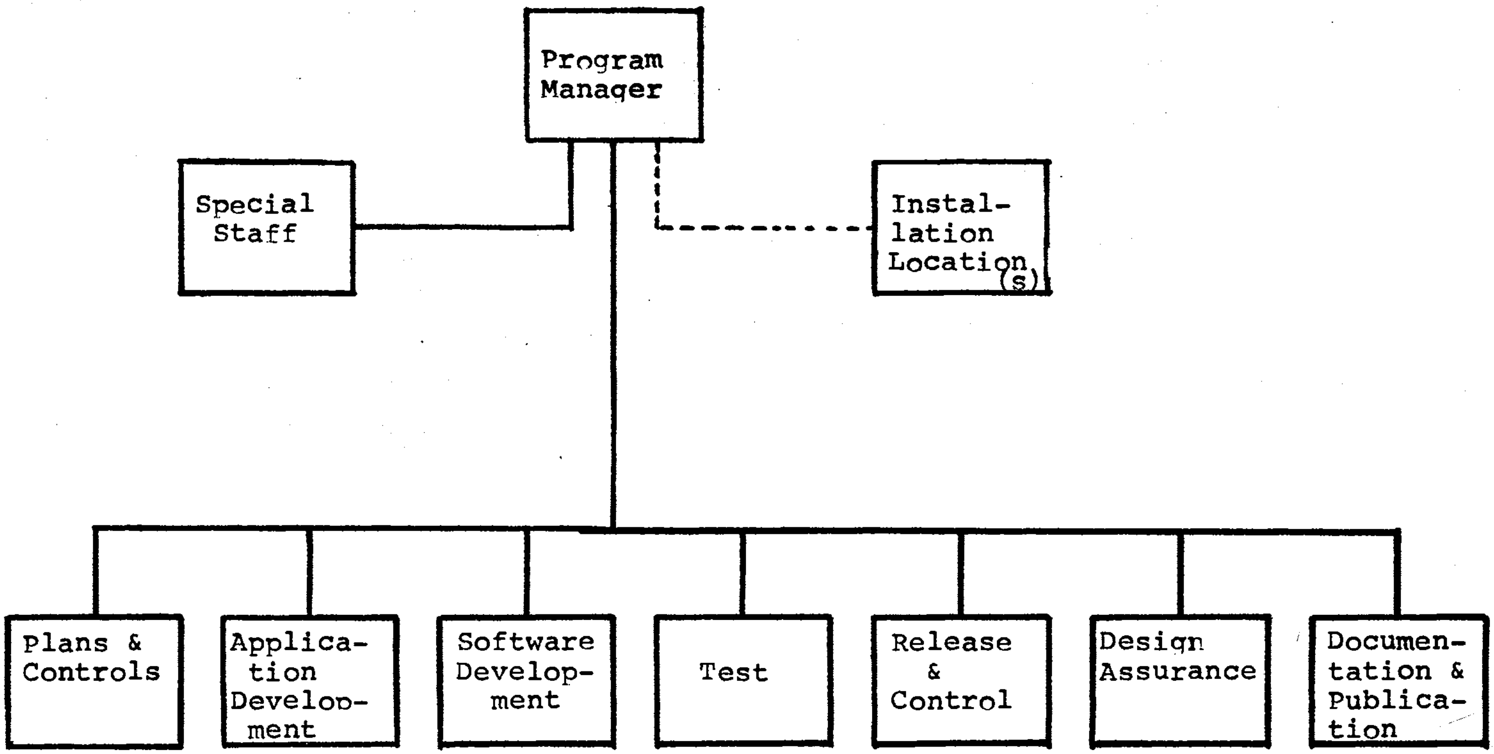


Figure 20. Management Information System Organization

teamwork and personal contact are what makes the technological worth of a management information system. The planning and control process must monitor budgets, schedules, and resources in each of the eight phases. Design assurance is primarily involved with the design and testing phases to insure that the system objectives, performance standards, and system specifications are met. Documentation must be published at the conclusion of each phase including the maintenance phase where enhancements are made. Application and software development organizations can be compared to the manufacturing organization. They build, debug, and insure installation of their programs and are the driving force to which all other organizations are really in effect staff. The test, release and control, and installation organizations add their support to the appropriate phase. Figure 21 depicts the complex information flow required for a large, real-time management information system.

Planning and Controlling Methodology

Management Information Systems just do not happen. They have to be carefully planned and then painstakingly controlled. There are a number of factors which make planning management information system a difficult process. For instance, there is generally a great number of variables and complex information which must be considered, there is a relatively high investment, the turnover rate for programmers is high, and there is a lack of historical data upon which to base estimates. If the project is to be successful, it must establish definitive project plans, periodically evaluate progress against those plans, and then have a mechanism for insuring corrective action. The quality is not something that will occur immediately. Design reviews from a technical,

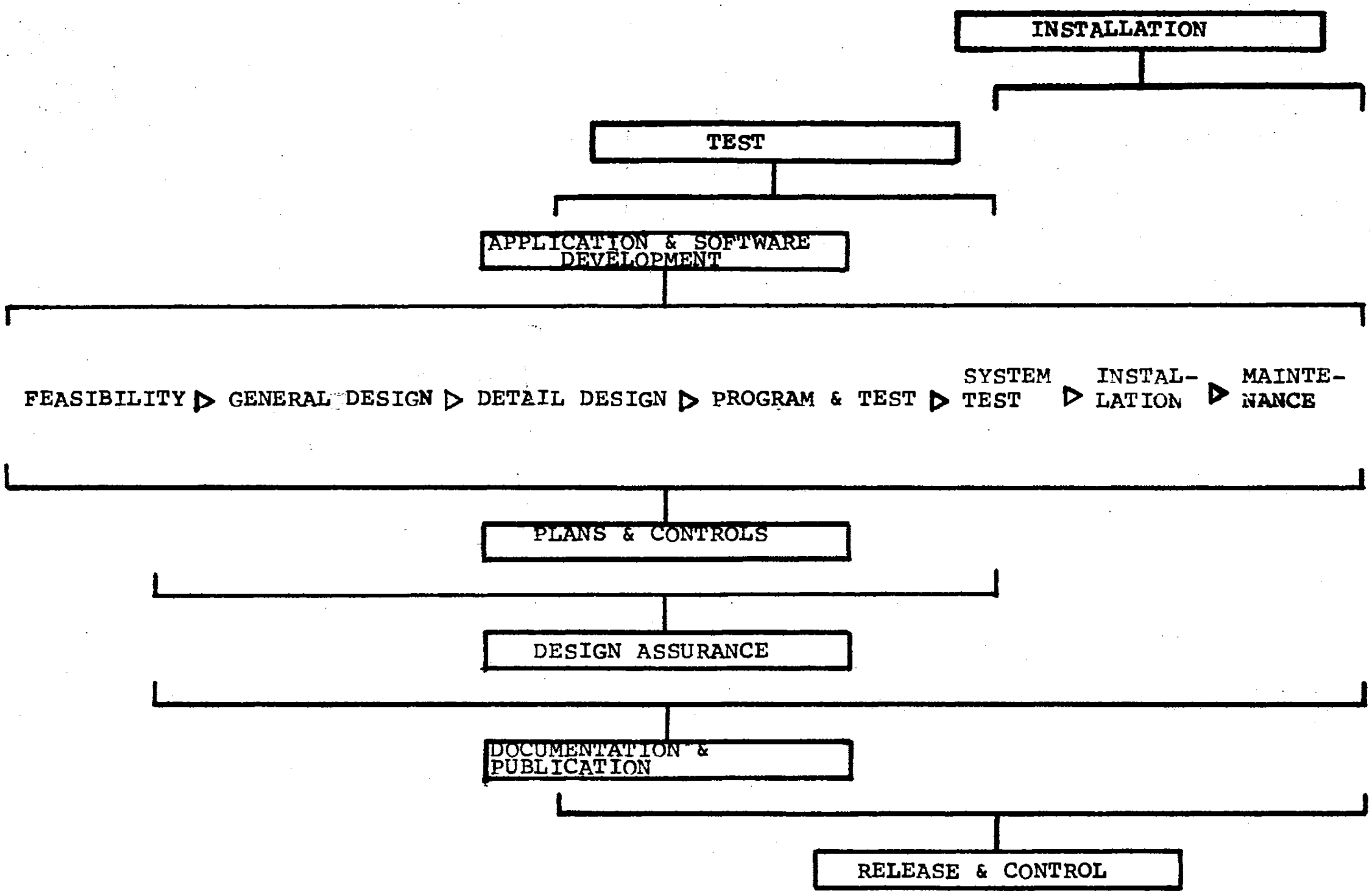


Figure 21. Management Information System Communication Flow

performance, documentation, and control perspective must be planned and completed.

Currently, underestimates for resource and time requirements are the rule rather than the exception. Nearly every program manager has had the experience of having a project miss commitment dates and overrun the planned resources. Even though this is a new area, more can and should be done to record, systematize, and generalize previous experience into future plans which will, in return, provide a stable base upon which realistic estimates can be made. This section provides a recommended methodology that will assist in estimating and then controlling resource and schedule requirements. It encourages consistency in the data and also explicit statement of the estimating factors used in the development of the resource and schedule projections. The inherent premise is that it is possible to identify and then quantify the variables in the various activities. This will provide a base for the minimum consideration and establish the methodology for which it is possible to state all the unique characteristics.

Many times the planners lose sight of the overall or general management considerations when analyzing the detailed activity complexities. Before looking into the detailed considerations, it must be pointed out that there are two types of loss factors which generally occur and should be applied over and above the detailed estimates. The first type of loss factor is termed a non-project factor. It includes the time that the personnel spend on rest periods, discussions of a non-business nature, coffee breaks, extended lunch periods, etc. This factor can range from five to ten per cent of the total time (18). The second type of loss factor is termed an off-project factor. It includes the time spent on activities not related to the project such as training, travel, accident, holidays, education, attrition, military leave, vacations, personal time, sickness, and special assignments of a non-project basis. This factor can range between twenty to thirty-five per cent of planned manpower (18). Therefore, if these two loss factors are considered together, there can be a loss from a low of twenty-five per cent to a high of forty-five per cent. Without consideration of these factors in sufficient magnitude, it is easy to see why many systems do not meet their initial commitments. It must always be remembered that the percentages will vary depending upon each unique project.

Another consideration which in many instances is not brought into proper perspective is the hiring and orientation activities. Enough elapsed time and resources should be planned for proper assessment of skill, recruitment, interviewing, selection, waiting, and orientation characteristics. Too often, it is wrongly assumed that a programmer can be hired immediately and will be fully productive in a week.

Although most management information systems follow the eight phases, there are many unique activities within each phase that are particular to each project. There is a multitude of considerations which must be planned. A few typical questions are listed as follows:

(1) How should it be documented?

(2) Who is to prepare the test data?

(3) When should design reviews be held?

(4) Where should the system be tested?

(5) When is the documentation needed?

(6) How will the schedules and resources be planned and controlled?

(7) What is the testing strategy?

(8) What is a design review?

(9) Will the user participate in test plans?

(10) How will all the programs be delivered to the user?

(11) How will changes to the installed system be handled?

To answer these types of questions and to understand the interrelationship among them, an activity network should be developed and then used. Basically, an activity network is a pictorial representation of the various activities and events which take place in order to meet predefined objectives. Each activity must be completely understood so that appropriate resources and schedules can be assigned. Within the activity network, an event is the start or completion of an activity. The circles (nodes) on the activity network are events. The lines and arrows represent activities. The arrows show that an activity cannot be started until its previous activity has been completed. An initiating and terminal event defines each activity. The events are only points in time. The activities require time and resources. When more than one activity begins at an event, more than one path is created and this is called a burst node. This occurs when more than one activity is being worked on at the same time. When specifying the various activities in the various phases, it is better to choose activities which have defined beginning and ending events. One item which lends itself to this approach is the documentation because it is a defined end product which either does or does not exist.

In order to insure that the activity network is representing the same information to all people, it is necessary to follow a consistent format in describing each activity. It is probably the lack of time spent on this phase of the planning which later causes the most problems in controlling the project. There must be a detailed base upon which to establish schedule and resource requirements. A suggested format is found in Table IV.

TABLE IV

ACTIVITY PLANNING AND CONTROL FORMAT

PHASE:	One of the eight portions or major parts of the overall process.
ACTIVITY NUMBER:	The node number as found on the activity network.
ACTIVITY NAME:	The short description found on the activity network.
ACTIVITY DESCRIPTION:	A concise statement of the activity with defined start and end points.
ACCOUNTABILITY:	The department name which has responsibility for insuring the output checklist for the activity will be completed by the scheduled date.
INPUT CHECKLIST:	A list of the items needed for the start of an activity.
OUTPUT CHECKLIST:	A list of items produced (end products) during the activity.
STAFFING:	A list of the job functions which could be required for the activity.
TASKS:	A definition of the work which needs to be performed.
ESTIMATING FACTORS:	The quantitative and qualitative values associated with the activity. They are intended to help identify the areas which affect the schedule, resources, or time estimates.
REFERENCES:	Any other information sources which would assist in accomplishing the activity or understanding its content.

The activity network is easier to implement if it depicts the actual plans rather than some theoretical plan which will never be utilized. In order to portray the actual plans, there needs to be considerable information flow between the technical and managerial personnel. Rather than passing over any issues which may arise as being trivial or superficial, there needs to be an action plan established for resolving them. In this process, the communication gaps are identified and corrective work is done to fill them. The activity network becomes a communication mechanism upon which all people are planning. It tends to foster a group directed goal upon which all management levels and technical personnel can see their contribution. All responsibilities, including the users, have been clearly identified and communicated so that there are no surprises.

After the activity network is developed and agreed to by appropriate management levels, it is necessary to obtain schedule and resource commitments. In all cases, the manager responsible for the completion of an activity must provide their own commitment. Never should a planning group be committing resources or schedules.

It must also be realized that the precision of the estimates is directly proportional to the amount of information available to the estimator. The finer the breakdown and understanding of the tasks to be performed, the better the estimate. One method for deriving estimates is known as the analogy method. To estimate by analogy the estimator should have an experienced background in the type of work being estimated. The estimator performs a factoring operation which estimates time and resources based on similar historical situations. The estimate is based on ideal conditions and then adjusted to allow for a particular project's loss factors, environment, and resource considerations. Many initial estimates based on ideal conditions are accurate as far as they go. The difficulty or inaccuracy of these estimates often lies in the fact that several important factors affecting the time duration or resource requirements were not considered. The estimator must have all the information provided on the input/output descriptions, task descriptions and estimating factors at his disposal. In this way negative and positive deltas against the original ideal estimate can be made. This represents the maximum insight which can be given to the planning process.

It must be remembered that a partial estimate is better than none at all. Time and resource estimating should be done with appropriate estimating factor first for each task, then activity, and finally each phase. This should be done independent of schedule constraints but should consider all resource constraints. The planning assumptions, including personnel identified to the various tasks, must be stated. The estimated considerations should be added to determine task, activity, and phase resource and time allocations as depicted in Figure 22.

The next step is to plot each task, activity and phase, on a time scale. This provides an overview of the first-pass schedule and resource requirements. Then the schedule commitments, if they have been made, are superimposed on the first-pass schedule. If the committed date is later than the first-pass date, it may be desirable to take people off the project. However, in most instances the situation is reversed. The committed date is earlier than the first-pass date. When analyzing the activities which lead to the critical commitment date, it will be evident that there is one or more sequence of events which takes

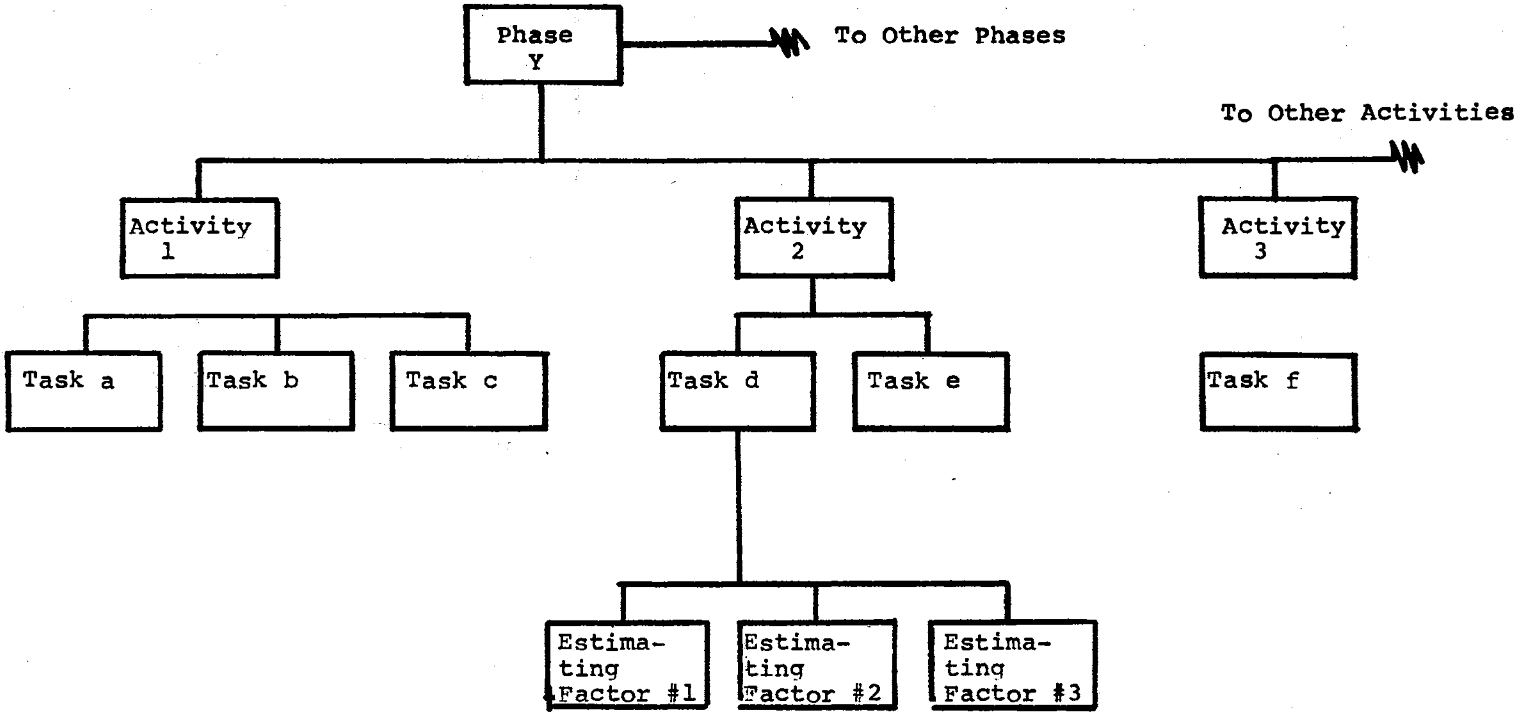


Figure 22. Hierarchy of Resource and Schedule Estimates

the most time. This is called the critical path and if the activities in this sequence slip, the whole project slips. In this case, the analysis has to determine if there is a better method to accomplish the desired goal, if more people can be placed on the critical activities, if overtime can be utilized, etc. The first-pass schedule is shown in Figure 23.

One of the questions which arises is - what is the value of the activity network after it has been developed? It has many uses after its development and acceptance. Each manager will probably utilize it in a slightly different manner depending upon his management style. If he has been part of the planning process, he knows that it provides a definitive base for the key events (many times called milestones). He is aware of the defined sequence of overall activities in reaching the key events. If there is another sequence of activities which is being followed, it should be so stated and the activity network changed. This will result in communication between the affected areas. In no case should the activity network dictate the only method which is to be followed. It is only a statement of plans and should be updated as plans change.

One of the big advantages of the activity network is its ease of application in the control process - especially the phase review concept. Since there is so much fluidity in development, testing, and installation activities, the program's resources and schedules should not be committed from the first day through the end of the project. There are too many items which change. User requests, new data processing techniques, management decisions, technology, economic environment, etc., are all variables. Inherent in the phase review concept is

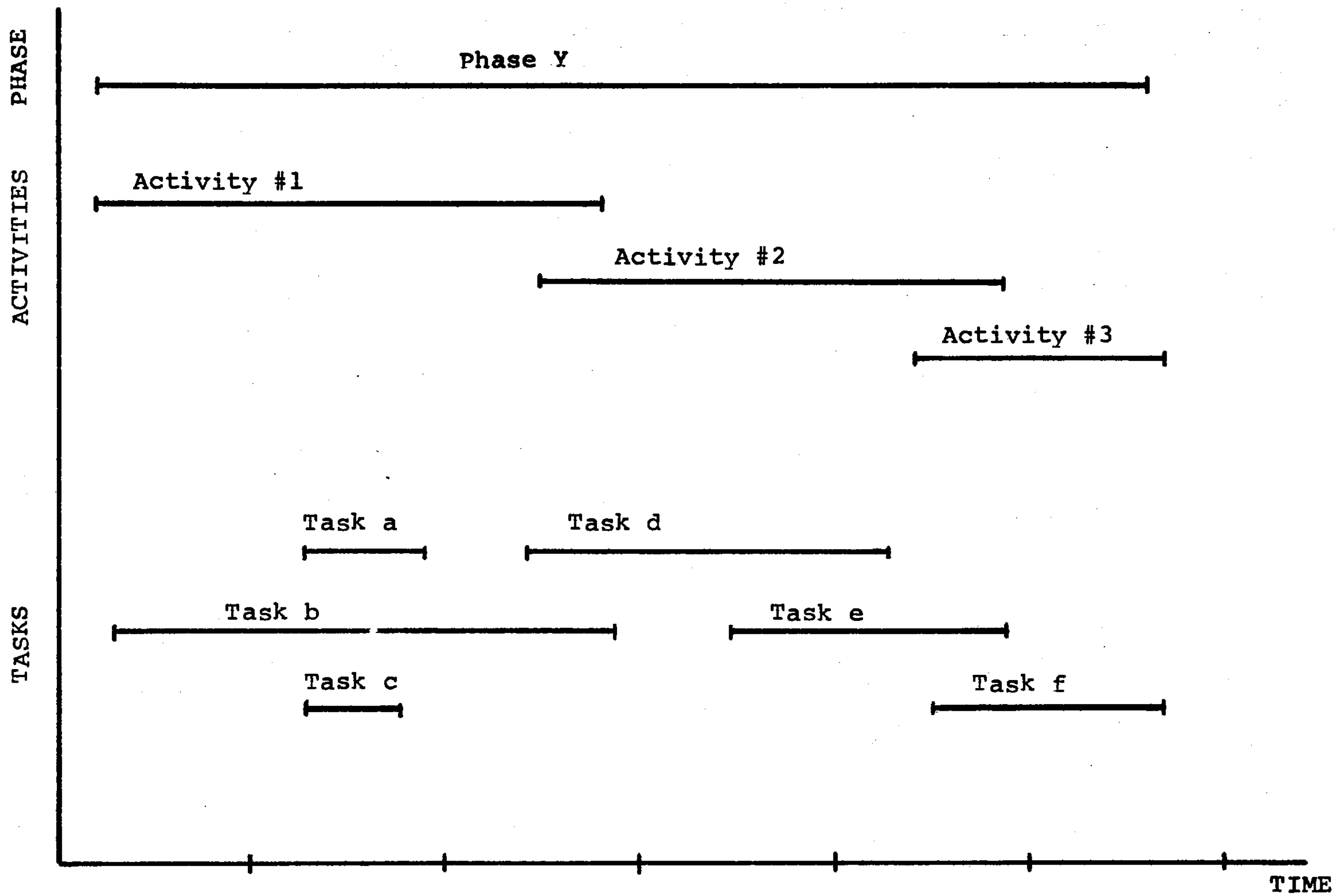


Figure 23. First-Pass Schedule

the resource and schedule targets for the following phases. As each phase is accomplished, the previously targeted resources and schedules for the next phase are revised as necessary and then committed. Also, at this time, the other remaining phases are analyzed again and, if necessary, new resource and schedule budgets are established. In this manner, a project can follow a sequenced set of phase reviews and provide more realistic cost and schedule projections all the way through installation. The basis for each phase review meeting will not only include the resource and schedule commitments, but also any other items which have been identified in the activity descriptions. It must be kept in mind that the end result of the phase review meeting is a GO or NO GO decision on the next phase. It also must be recognized that a good manager will not just have his project work up to a phase review meeting checkpoint and stop there and await the results of the meeting. The manager will be working on activities in the next phase. This management strategy represents a risk. At all times the manager should attempt to keep the amount of work on the following phases to a minimum and thereby keep the risk at a minimum.

It must be pointed out that any method of estimating resources and schedules, whether based on intuition or a more formal technique, is no better than the knowledge, experience, and judgment of the estimator. The planning and control concepts presented should be regarded as a supplement to aid in this judgment rather than a complete replacement for it. The estimator will probably develop formulas for estimating factors, under certain conditions, to provide more quantitative means of deriving the resource and schedule estimates. However, the experienced estimator will sense when these unique conditions are in existence and supply temperance judgment to obtain a more realistic value if necessary. Variables which will have real significance but are not quantifiable are such things as: initiative, desire, and intelligence. However, these qualitative factors can be applied to the derived estimates to obtain the best possible resource and schedule commitments.

One of the most common problems in over running schedules and resources is that not enough time was taken to control the program. A program can have the best plans for reaching an objective, but unless proper controls are utilized, the program will fail. Every program has to have a central control mechanism which is clearly visible to management, the installation location personnel, programmers, system analysts, and other program members. The plans and controls department usually will have the control function. They will have the one central repository of information concerning the program's status. Also, they should have a standardized procedure to handle the multitude of problems which arise. This may take the form of a critical problem list which will serve as the central communication mechanism for helping to resolve problems.

CHAPTER IV

MANAGEMENT INFORMATION SYSTEM MODEL DEVELOPMENT

Modeling

There are many different types of models. For instance, it is possible to write a set of equations which would describe a system's characteristics. This is called an analytical model. It is also possible to actually make a physical representation of the system being considered. This is done many times in building a scale model of an airplane and experimenting with it in a wind tunnel. Physical and analytical models are quite different. The physical model is something which is concrete and can be physically manipulated while the analytical model is very abstract. However, there is also a type of model which falls in between these extremes - the descriptive model. It uses symbols to represent the activities of a system. This model portrays the various interactions between the variables, constants, and constraints in the system. This is the type of model which is used in this dissertation. In all definitions, the term "model" is defined as a representation of the real or actual situation. In order to adequately portray the actual situation, it must adequately contain the cause and effect relationships.

A simulation model represents a unique type of model. When one can change the model to analyze various alternatives, without changing the real situation, a simulation is being performed. There are many different kinds of simulation techniques which can be used in manipulating the model. Simulation can be accomplished through techniques ranging from paper and pencil through computers. This dissertation uses the term simulation model to include the computer in the manipulation of the model. Therefore, it is necessary to understand the usefulness of computer simulation languages.

Decisions based on many complex interrelated activities are not easy to make. These decisions usually center around planning under severe environmental constraints and can represent substantial capital. It is also likely that there is a lack of previous identical projects upon which it is possible to base decisions. If it were possible, the least risky decision-making practice would be to adopt a future plan, watch it in operation, and then make various repetitive refinements until the desired result is achieved. This involves a high capital investment and also a long elapsed time duration. In this type of situation, a realistic approach is the creation of a model, which represents the actual situation, and then the manipulation of the model to evaluate alternative solutions. When one achieves the alternative which maximizes the ratio of benefits divided by burdens, the recommended solution has been achieved.

There are many diverse attitudes toward computer simulation. A few of the more important considerations in deciding to use computer simulation are provided in Table V.

TABLE V

ADVANTAGES AND DISADVANTAGES OF COMPUTER SIMULATION

Advantages	Disadvantages
1. Provides insight into complex situations	1. Has a high cost
2. Can experiment on various alternatives in a short time period	2. Requires expensive computers
3. Can easily manipulate complex situations	3. Can require long elapsed time duration before model is constructed
4. Do not have to change actual situation	4. May require many iterations before desired alternative is chosen
5. Provides means of training new personnel	5. Must represent the real situation
6. Can compress future events into brief time intervals of present time	6. Requires experienced personnel
7. Does not require expensive commitments to actual plan	7. May be difficult to use various computer languages
8. Is accurate and never makes a mistake	8. Will not provide an actual solution but will provide various alternatives
9. Has a wide variety of useage	9. May be difficult to match language versus computer

Model Considerations

There are various general model criteria set in place. Rather than provide a detailed description of each one of the criteria, they are listed in Table VI and then a short description of the more important criteria is provided.

TABLE VI

MODEL CRITERIA

- o Must be easy to construct
- o Must provide comprehensive output reports
- o Must have simple input requirements
- o Must process data in a quick manner
- o Must be an on-line system
- o Must be manipulative
- o Must be able to portray actual situation
- o Must aid in planning and scheduling
- o Must assist in providing better communication
- o Must provide progress to plan reports
- o Must provide benefits
- o Must provide a capability for planning, replanning, and progress evaluation

One of the most important criteria is that the model must be able to portray the situation which is being studied. If the model cannot adequately do this, the resultant benefits will disappear and in fact there will probably be a cost in weighting the expected benefits versus the expected burdens. However, once the model does represent the desired situation, a method for changing the model to analyze various alternative solutions can be derived.

Another important criteria is that the model must be constructed so that it possesses on-line and real-time characteristics. With this in mind, the model can be changed and the various alternatives immediately sized. This allows for an interaction between the user and the model. By utilizing this approach, the user will use the model so that it will assist in the decision-making process and also will enable a better model to be developed.

The last important criteria mentioned here is to have a model which can be used easily. The model must be easy to build and then to modify for analyzing various alternative. This will encourage many users to utilize the model for helping in the decision-making process. This will, in turn, help to keep the model up-to-date (current) so that it reflects the actual situation.

Computer Languages

Analysis of multiple alternatives would not have gained its importance today if it were not for the computer. The computer had made it economically possible to run analysis of large, complex situations. This has only happened over the past fifteen years with the use of enhanced computer languages and also the larger computers. Today, it is possible to use various types of computer languages which are easily understood by both the computer and the people who are using it, to meet their objectives.

There are many different computer languages which can be used. Each has its own advantages and disadvantages. For instance, it would be possible to use the IBM General Purpose Simulation System in discrete system simulation. Or, it is possible to use the IBM continuous system modeling program in continuous system simulation. The choice of which one of the two languages which should be used is dependent upon the complex interrelationships which characterize the model. The continuous system simulation takes a broader view of the system than the discrete system simulation. The entities and events which would be treated as discrete items in the discrete system simulation are aggregated into continuous time processes in continuous system simulation.

There are many other alternative simulation languages which may be considered. It is possible to build operations research (O.R.) models of various real world situations by using various O.R. oriented languages. An example of this would be the utilization of the IBM Mathematical Programming System in solving an Oil Corporation's large, complex linear programming problem. Another type of simulation language which must be considered and which falls into the area of project management is the IBM Project Management System. It assists in the planning, replanning, and controlling of projects.

Each one of the computer languages - whether discrete system, continuous, operations research, project management - have their strong and weak points. However, the type of simulation technique selected must best match the model criteria as listed in Table VI. After analyzing each computer application against these criteria, it becomes evident that a project management simulation technique is best.

After deciding to use a project management simulation language, it is necessary to select the one which also most closely matches the desired model criteria (Table VI). There are many different languages which are available (19). A few of the possible choices are listed in Table VII.

TABLE VII

VARIOUS PROJECT MANAGEMENT COMPUTER LANGUAGES

Program Name	Computer	Approximate Capacity (Activities)	Real-Time	Resource Allocation	Bar Charts	Cost Control
PERT	Control Data 1604	3,000	No	No	No	No
Time-PERT	Burroughs B5500	524,288	No	No	Yes	No
CPM/MONITOR	General Electric 400/600	2,000	No	No	No	No
PERT	RCA 501	2,000	No	Yes	No	No
1130 PCS	IBM 1130	2,000	No	No	Yes	Yes
MINIPERT	IBM 360	200	Yes	Yes	Yes	Yes
PMS/360	IBM 360	Unlimited	No	Yes	Yes	Yes

After looking at the various project management computer languages which are available and comparing these to model criteria stated in Table VI, MINIPERT (9) was selected. It is an interactive (between user and computer via an on-line, real-time terminal), conversational project management technique. It operates under an APL/360 system configuration and uses terminals as the input/output device. It requires an IBM System/360 Mod 40 or larger.

Model Development

The emphasis in developing a model of a management information system is on the testing activities. However, it is necessary to also understand all the phases and be able to relate the testing activities to this interrelated framework. Therefore, the general model treats all of the eight phases in an integrated manner with special emphasis on the programming, system test, and installation phases.

All of the activities in the general model have been developed and defined so that they are generic. Therefore, they are universal and can be applied to the development, test, and installation of any large, complex, real-time management information system. Special emphasis has been placed on the development of a "testing philosophy". It is portrayed in the general model network found in Figure 24. The activity descriptions of the network are provided in Appendix A. The planning and control methodology as described in Chapter III is used as the base for both the network and the activity descriptions.

The first item that must be examined in the development of a management information system general model is the relationship between the eight phases assumed to constitute it. Although it is possible to distinctly break the various phases through detailed planning, in reality it is almost impossible to develop the MIS in this manner. In effect, risks must be taken. Follow-on phases must be started before the completion of the previous phase. These considerations are depicted in the general model, where the overlapping phase characteristics are portrayed. However, as pointed out in Chapter III, the amount of work which is undertaken on follow-on phases must be minimal. This is so

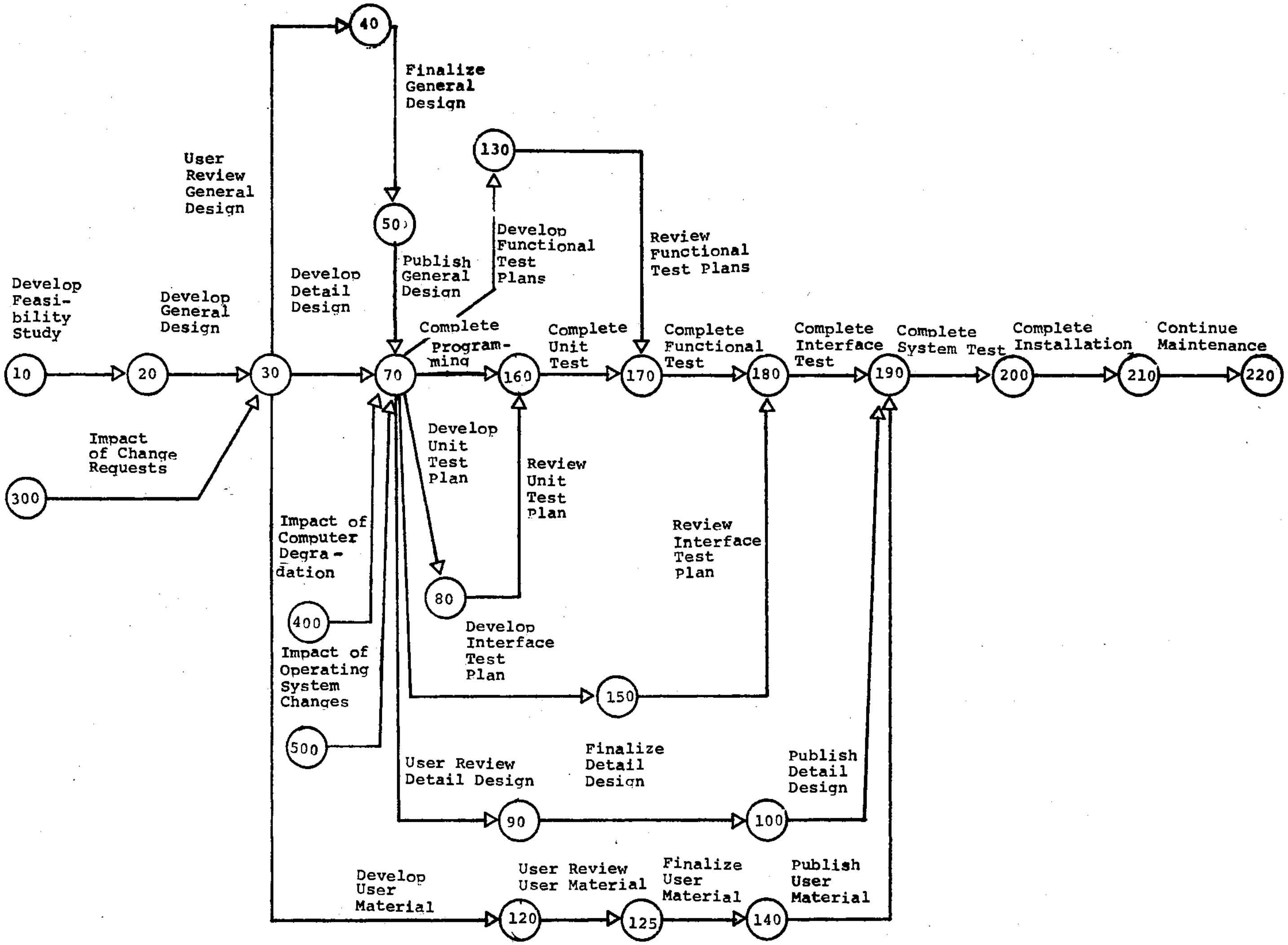


Figure 24. General Model of a Management Information System - Emphasis Testing

that if the phase review meeting decides to terminate the project the loss will be minimized.

The major emphasis in the activity network, provided in Figure 24, concerns the testing activities. Each activity has been defined and integrated with all other activities so that the testing objective can be successfully accomplished. The testing objective in this dissertation is stated as follows: There must be a structured, non-redundant, series of progressive tests which insure that the management information system will be installed in the shortest amount of elapsed time with a minimum resource level. The implementation of this philosophy with regard to the various activities follows a set of testing axioms as stated in Table VIII.

The axioms in effect assist in defining each of the activities in the programming and test, system test, and installation phases in a rational consistent manner. It is important that all areas are aware of the testing axioms so that there can be common communication on why the testing is being done in a given manner.

The next step is to define each activity. The level of definition is difficult to determine; however, it should be general enough to provide flexibility and at the same time definitive enough to be meaningful. The definitions used to support the general model of a management information system (Figure 24) are located in Appendix A.

Every attempt has been made to insure that the general model is representative of a large, real-time management information system. After the activities are structured and defined to insure the successful accomplishment of the testing axioms, it is necessary to apply

TABLE VIII

TESTING PHILOSOPHY AXIOMS

Axiom
o The MIS users must review test plans to insure that they understand their makeup.
o There must be a philosophy and procedure for handling changes to the programs.
o There must be a planning and controlling methodology for computer availability.
o There must be an established set of standards against which the management information system can be measured (for milestone completion, operating performance, etc.).
o There must be a hierarchy of testing so that redundant testing can be avoided and also to help insure quality.
o There must be a communication and progress reporting mechanism across all functions.
o There must be appropriate user documentation and education.
o Testing at the installation location must be minimized to insure the most economical testing.
o A common test data base must be established for all programs for all testing.
o All the programs must be tested at one central location whether the personnel are located at the facility or whether they are remote to the facility.
o There must be a central repository for all data elements, all records, and macros.
o There must be predetermined test results against which actual results are compared.
o There must be identification of responsibility from start to finish.

representative time durations. A survey was taken of various IBM management information system projects which lead to the data in Table IX.

TABLE IX

PERCENTAGE OF ELAPSED TIME PER PHASE

Phase	Percentage of Elapsed Time
Feasibility Study	4.0
General Design	8.0
Detail Design	14.0
Programming and Test	35.0
System Test	20.0
User Material	8.0
Installation	11.0

The percentages in Table IX were used to apply representative time durations to the various detail activities in the general model network. It should be noted that the maintenance phase is not included in Table IX. Maintenance is a continual activity until a predefined point is reached when it is discontinued. A list of the general model data input, including the time durations, that was finally developed is listed in Table X (is actual MINIPERT input list).

The development of the final general model as portrayed in Figure 24 was not a one time effort. There were many iterations in balancing

TABLE X

GENERAL MODEL DATA INPUT

COUNT	PRED	SUCC	DUR	SCH DATE	ACT DATE	DESCRIPTION
1	20	30	12.00			DEVELOP GENERAL DESIGN
2	30	40	10.00			USER REVIEW GENERAL DESIGN
3	30	70	35.00			DEVELOP DETAIL DESIGN
4	30	120	20.00			DEVELOP USER MATERIAL
5	40	50	10.00			FINALIZE GENERAL DESIGN
6	50	70	10.00			PUBLISH GENERAL DESIGN
7	70	80	20.00			DEVELOP UNIT TEST PLAN
8	70	90	10.00			USER REVIEW DETAIL DESIGN
9	70	110	70.00			DEVELOP SYSTEM TEST PLAN
10	70	130	30.00			DEVELOP FUNCTIONAL TEST PLAN
11	70	150	40.00			DEVELOP INTERFACE TEST PLAN
12	70	160	43.00			COMPLETE PROGRAMMING
13	80	160	10.00			REVIEW UNIT TEST PLAN
14	90	100	10.00			FINALIZE DETAIL DESIGN
15	100	190	10.00			PUBLISH DETAIL DESIGN
16	110	190	10.00			REVIEW SYSTEM TEST PLAN
17	130	170	10.00			REVIEW FUNCTIONAL TEST PLAN
18	140	190	10.00			PUBLISH USER MATERIAL
19	150	180	10.00			REVIEW INTERFACE TEST PLAN
20	160	170	15.00			COMPLETE UNIT TEST
21	170	180	15.00			COMPLETE FUNCTIONAL TEST
22	180	190	15.00			COMPLETE INTERFACE TEST
23	190	200	50.00			COMPLETE SYSTEM TEST
24	200	210	28.00			COMPLETE INSTALLATION
25	210	220	0.00			CONTINUE MAINTENANCE
26	300	30	0.00	5/02/73		IMPACT OF CHANGE REQUESTS
27	400	70	0.00	5/15/73		IMPACT OF COMPUTER DEGRADATION
28	500	70	0.00	6/04/73		IMPACT OF OPERATING SYSTEM CHANGES
29	125	140	10.00			FINALIZE USER MATERIAL
30	120	125	10.00			USER REVIEW USER MATERIAL
31	10	20	10.00	4/02/73		DEVELOP FEASIBILITY STUDY

the testing axioms, testing objective, activity relationships, and activity durations. After each run the output reports were analyzed to determine if they met the testing objective. If it did not accomplish the objective, it was rerun and reanalyzed. This was done until the general management information system model was developed with the data input as listed in Table X. One of the noteworthy aspects of the entire model development process is that the reports generated from MINIPERT are real-time (within 5 seconds). There was no need to wait for the batch processing of data. This helped in debugging the model in that the effect of making modifications could be seen immediately. This conversational mode saved many hours of waiting time.

There are various reports which can be generated from MINIPERT. A list of the various reports and a statement of whether or not they are in Appendix B is found in Table XI. All the reports in Appendix B are based on the general model data input as provided in Table X which is, in turn, based on Figure 24. It is important to know that the reviewing of test plans and also the preparation of documentation are either on or near the critical path. This is because the programmers actually performing the various tasks do not generally allow adequate time for these activities.

TABLE XI

MINIPERT REPORTS

Report No.	Description	In Appendix B
0	Sorted by order of input	No
1	Sorted by predecessor	No
2	Sorted by successor	No
3	Sorted by duration	No
4	Sorted by ES (Early Start date)	Yes
5	Sorted by EF (Early Finish date)	No
6	Sorted by LS (Late Start date)	Yes
7	Sorted by LF (Late Finish date)	No
8	Sorted by slack	Yes
9	Sorted by first three critical slacks	No
10	Those activities presently in progress	No
11	Activities to become due within 30 days	No
12	Activities to have been completed by today	No
13	Sorted by description	No
14	Sorted by predecessor-successor	Yes
-	Activities in a network format	No
-	Activities in a network format with title and status	No
-	Bar chart of Report Requested before	No

CHAPTER V

MANAGEMENT INFORMATION SYSTEM MODEL APPLICATION

Description of Actual Situation

The application of the general model developed in Chapter IV is made to an IBM Common Systems Development project. The project selected is called a Common Manufacturing Information System (CMIS). It is designed to replace much of the current data processing activity of the thirteen manufacturing facilities in North America and the manufacturing facilities in IBM World Trade locations throughout the world. It has an integrated data base which supports a wide variety of operating functions which are required to control the manufacturing activities. This includes everything from supporting multi-plant operations planning to assisting in the controlling of materials logistics. It is a real-time, on-line system which can respond to events as they occur.

The integrated data base contains records of essential information. This insures that there is a standard format of data between the various user locations. This one-source data bank information provides the paths of information flow among the various plant locations and division headquarters through a leased-line communication network.

The integrated data base contains records of essential information. This insures that there is a standard format of data between the various user locations. This one-source data bank information provides the paths of information flow among the various plant locations and division

headquarters through a leased-line communications network.

The IBM CMIS is being developed at various locations throughout the United States. Table XII provides a description of the function and its location.

TABLE XII

CMIS FUNCTION AND LOCATIONS

Location	Function
Kingston, N. Y.	Subsystem which maintains the basic manufacturing records.
Endicott, N. Y.	Subsystem which explodes the product schedule into orders required of each plant.
Rochester, Minn.	Subsystem which handles the release and control of internal production orders, the determination of components required for those orders, and the allocation of available components.
Poughkeepsie, New York	. Subsystem which provides information on vendors. . Subsystem which maintains perpetual inventory of parts in the warehouse. . Subsystem which supports the final assembly of products to customer order providing customized assembly parts lists and assembly instructions for each machine. . Subsystem which includes programs to handle file access, teleprocessing control, checkpoint restart.
Raleigh, N. C.	Subsystem which supports retrievals and inquiry.
Sterling Forest, New York	. Centralized testing activities. . Centralized release and control activities. . Centralized design assurance activities. . Centralized planning and controlling activities. . Centralized documentation activities.

As one might visualize from this description, IBM's CMIS is a large, complex, real-time management information system which is being installed at locations throughout the world.

Model Modifications

It is not possible to take the general management information system model and apply it to a specific situation without modifications. Each MIS has its own unique requirements. Therefore, the degree of modification will be dependent upon the unique application and the user's requirements. This section discusses the modification to the general model which were required so that it could be applied to IBM's Common Manufacturing Information System. It serves as an example of how it was applied so that others wishing to apply the general model will have a base upon which to work.

The first item which becomes quickly apparent when starting to apply the general model is the degree of complex detail and also the technical expressions concerning the specific management information system. However, if enough time is taken to understand the terminology and the interrelationships of activities, it is possible to begin developing a strategy for applying the model. Therefore, step number one is to understand all the characteristics of the specific management information system. This was accomplished by the writer having jobs in different aspects of the management information system over the past four years.

The next step is to insure that each activity for the specific model is stated in precise detail. Activity 70-80, 80-160, and 70-160 have been chosen from the general model to serve as examples. These activities are applied to the IBM CMIS project within the concepts and strategy of the general model. Activity 70-80 is broken into three activities, activity 80-160 is broken into three activities, and activity 70-160 is broken into eleven activities. This demonstrates that there is a substantial amount of work required for modifying the general model. It also shows that there is a need for a detailed understanding of the specific management information system as well as the general model. See Figure 25 for an example.

Although each activity in the general network has to be intensely sized for application to the specific management information system, there are three activities which historically deserve the most attention and will change more rapidly than any of the others. These are impact of change requests (activity 300-30), impact of computer degradation (activity 400-70), and impact of operating system changes (activity 500-70). Because of their particular importance, the modifications required will be discussed in more detail.

Change requests are a continual concern in any management information system. They may arise from various sources. The various users, programmers, testing personnel, etc., may create a request for changing program code based upon many factors. They may be based on performance improvements, maintenance requirements, or user operating procedures. Depending upon the magnitude of the change request and the relative status of the management information system, the change request can take on different degrees of criticality. For instance, if a major change request is required and the MIS is in general design, the change can usually be made with minimum effort. However, if a major change request is required and the MIS is just about to finish testing, there

FROM GENERAL MODEL

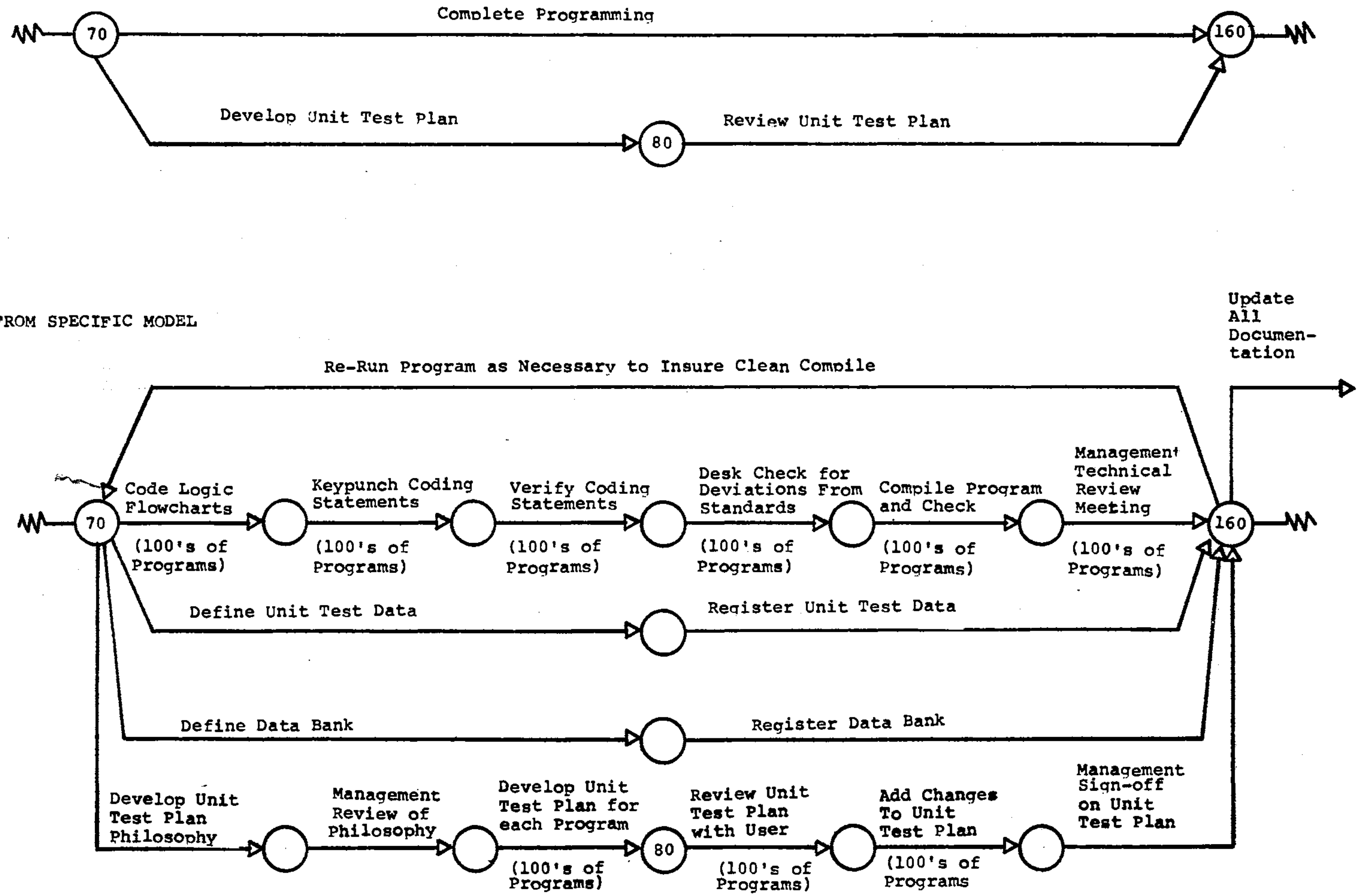


Figure 25. Comparison of General to Specific Model

are usually substantial problems. Change requests generally have to be handled on a one-for-one basis depending upon their magnitude and degree of development and testing which has been done. The actual placement of the change request activity in the detail network will be dependent upon the specific change request and the relative status of the specific management information system. The change request activity must be placed at a point in the detailed network where it constrains the future activities in the same time frames and logical interrelationships as in the actual situation.

Another factor which is of concern is computer degradation. The results of computer degradation are easy to measure - poor turnaround. Turnaround is defined to mean the elapsed time from when a programmer submitted the job to the time when the programmer receives the job. Computer degradation can be caused by hardware, software, or operations problems. It is considered by many to be the most critical factor contributing to schedule slippages and additional resources. However, most of the programming managers are quick to point out the obvious problems and not quick enough to identify other alternatives. When the computer system is either not working or not turning the programs around quickly enough, the programmers could be working on documentation, test plan reviews, desk checking, etc. However, there are key steps which must be undertaken to insure that these problems are minimized. This begins with the common definition of terms used in measuring key processes and also communication between all areas. As in the change request activity, the computer problems become more important at the later stages of test. If the MIS is just about to finish test and the computer becomes degraded, the impact is much greater than if test is just beginning.

The actual placement of the computer degradation activity in the detailed network is straightforward but the degree to which it should be made to constrain the future activities is a difficult judgment. To make it even more difficult to adequately size the impact, the computer degradation is random and varies in degree from being totally inoperable to being slightly inconvenient. However, the judgment must be made and the detailed network changed accordingly.

The last factor to be discussed is the changes to the operating system (O/S) which impact the testing of the application programs. Again the magnitude of the problem is a function of the management information system progress toward installation. The changes which are being made to the operating system have to be identified and then communicated to the appropriate personnel. There are very few instances where a change to the operating system does not affect the application programs. As with the impact from computer availability, it is extremely difficult to know how to assess the impact of operating system changes. At one moment the change may seem minimal and at the next second the change may be major or vice-versa. The emphasis should be on sizing the change, communicating it to all concerned areas, and showing in the detail model the projected impact of the change on schedules, resources, etc. The placement of the activity, reflecting the operating system change, in the detail network is dependent upon the status of the management information system.

Overall, the detail model had many more activities than the general model but followed the concepts and strategy of the general model. Where the general model would contain two activities for accomplishing a particular objective, the detail model could contain ten activities.

The only activities which were placed in a different constraining location than portrayed in the general model were those mentioned - the impacts due to change requests, computer availability, and operating system changes. This was necessary to accurately reflect the current status of the management information system.

Analysis and Validation

The detail model was continually being analyzed for improvements and also continually being validated against expected results as the management information system progressed through the various phases. This process resulted in modifications to the general model which are reflected in Figure 24. It also provided for progress control of the development, test, and installation activities.

The detail model was continually run and rerun to analyze the effects of alternate decisions upon the resources, schedules, and activity statements before they had to be committed to implementation. There was continual attention to insure that the detailed model and, thereby, its related output represented the actual management information system. This caused continual replanning and progress evaluation to insure consistent trade-offs between the schedules, technical characteristics, and resources. The MINIPERT reports assisted in leading to a precise, quick analysis of potential problem areas (see Appendix B for selected reports).

The detail model was validated each time an activity was completed (progress to plan). Since each activity had a statement of the end product(s) (found under the output checklist in the activity description), a schedule, and a resource commitment, it was a straightforward process to insure that the end products of the activity had been completed on time with the committed resources. It must be noted that there were activities which were completed earlier or later than scheduled and with either more or less resource. However, it was always possible to validate the activity by comparing it against the predicted criteria. The final and most important validation occurred when the management information system was cut-over to actual production. Again, the committed schedules, resources, and end products could be compared and validated to the actuals.

CHAPTER VI

SUMMARY, RECOMMENDATIONS, AND CONCLUSIONS

Summary

There does not seem to be a visible stopping point for the information processing revolution. The computer has been developed through three generations and today its application is only dependent upon man's imagination. Its application in organizations has caused some perplexing challenges. Some argue that the data processing function should be centralized while others argue that it should be decentralized. The same argument is taking place with regard to the planning and control function. Today, the tenents of centralization are winning because of economic savings and technical breakthroughs allowing centralization.

Today, most management information systems do not meet their predefined goals on schedule or within planned resources. The problem has not been in the "technical aspects" of the system, but rather in the planning and control process of answering the basic questions of how, who, where, why, and when. These problems can occur in any one of the complex set of interrelated activities which constitute the management information system. However, the problem is usually intensified if it must be corrected in the expensive testing activities. Historically, the problem has been viewed as being independent of all other activities. The emphasis has usually been placed on the activity in trouble, many times at the expense of the other activities.

Most management information systems progress through various overlapping phases. If one compares these phases between different management information systems, a similarity will evolve. This commonality begins with the feasibility study and continues through general design, detail design, programming and test, user material, system test, installation, and maintenance phases. In insuring a successful complex, real-time management information system, one must have an organization with effective communication flow, a planning and control methodology, and technical knowledge.

Next, a computer-based, general model of a management information system is developed. An IBM conversational project management application - MINIPERT - which uses terminals as the primary input/output devices, is used to construct the general model. The general model includes the eight phases assumed to constitute a management information system but emphasizes the testing phases - programming and test, system test, and installation. The general model activities are developed to compliment the dissertation's testing axioms while at the same time insuring that the testing objective is met. The testing objective is that there must be a structured, non-redundant, series of progressive tests which insure that the management information system will be installed in the shortest amount of elapsed time with a minimum resource level.

The general model is applied to an IBM Common Systems Development Project - Common Manufacturing Information System. It is not possible to apply it without modifications because each system has its own unique requirements. For every general model activity, the specific model has several activities. However, the general model always provides the concepts and strategy upon which the specific management information system, and thereby the specific model, is molded. The specific model, and thereby the general model, is validated by insuring that the input checklist and output checklist are met. Based on the general model application, there were changes which had to be made in the logic.

There are several conclusions which can be made from this dissertation. First, it is possible to develop a computer-based, general model of a management information system which emphasizes the volatile, expensive testing aspects. Next, it is possible to successfully apply the general model to a specific system. Also, it is possible to apply the dissertation's testing axioms to meet the test objective. Lastly, if the dissertation's recommended planning and control methodology is followed, the basic questions of how, who, where, why, and when can be answered and, thereby, the likelihood of meeting schedule and resource commitments is much higher.

Conclusions

The dissertation demonstrates that it is possible to develop a computer-based, general model of a management information system which emphasizes the volatile, expensive testing aspects. The general model consists of eight phases which includes a complex set of interrelated activities, and is independent of any particular management information system. It generalizes previous experience by recording and systematizing the various activities inherent in a successful management information system.

The dissertation also demonstrates that it is possible to successfully apply the general management information system model to a specific management information system. The specific system is an IBM Common System Development Project - Common Manufacturing Information System - which is being implemented at various locations throughout the world. There are modifications required to the general model so that it is tailored to the characteristics of the specific application. Whereas the general model is independent and contains generic activities, the specific model is dependent upon the particular system and contains actual activities.

Another important conclusion is that it is possible to apply the testing axioms proposed in this dissertation to meet the following test objective - there must be a structured, non-redundant, series of progressive tests which insure that the management information system will be installed in the shortest amount of elapsed time with a minimum resource level. The thirteen testing axioms ranged from insuring that the users review test plans to having one central test facility where all testing is done. Their validity was proven when the general model was applied to the specific system. In all cases, the specific model activities were defined so that they did not conflict with the testing axioms.

If the recommended planning and control methodology in this dissertation is followed, the basic questions of how, who, where, why, and when can be answered and, thereby, the likelihood of meeting schedule and resource commitments is much higher. This is accomplished by providing an approach which insures the development and analysis of alternative solutions, the establishment of better communications, and the development of continuous, progress reports. There are three areas which are potential problem areas and were given special attention -

change requests, computer degradation, and operating system enhancements. Analysis of how to react to these three problems is always made against the given set of technical objectives, the committed schedules, and the committed resources. The specific model shows that there should not be total concentration on one troublesome activity at the expense of the other activities. Activities are all interrelated and the change in one causes corresponding changes in the others. One of the major advantages in the recommended approach is that the management information system quality should be improved by insuring that everyone (managers, programmers, etc.) is working together as a team. They will be working together on one commonly understood base (network drawing) with all the constraints visible to meet the committed objective.

Recommendations

As time progresses, there will be larger capacity computers, faster computers, and better programming languages. There will also be better ways to design, test, and install the increasingly complex management information systems. As these changes occur, it is recommended that the general model developed in this dissertation be updated and fine-tuned to reflect the lastest state-of-the-art.

The general model elapsed time and resource data was gathered from experience within International Business Machines. It is recommended that this data also be gathered from other management information systems. This would insure that the general model would more equitably represent the management information system phases - feasibility, general design, detail design, programming and test, user material, system test, installation, and maintenance. When it is possible to more accurately define each phase's data, it becomes easier to relate and develop estimates for each activity inherent in the phase.

Another recommendation is that more work be done on the non-testing phases and then this work be added to this dissertation's general model. The phases where more work should be done are the feasibility, general design, detail design, user material, and maintenance phases. After completing this work, the activities should be added to the general model in the same manner that the testing activities were added in the programming and test, system test, and installation phases. If this were done, the general model would be more complete and would be of more value.

There are many management information systems which are being designed, tested, and installed by various developers. It is recommended that the general model be applied to these management information systems. In this way, the general model could be enhanced to reflect other ways of doing business while at the same time undergoing additional validation. This would increase the usefulness and make the general model more applicable to other developers.

The last recommendation, which is probably one of the most important, concerns making the general model (and thereby project management) work successfully. It is not possible to simply apply the general model to a specific management information system and sit back and wait for the benefits. There must be capable personnel together with adequate resources (such as computer time) allocated to the general model. There must be positive management attitude and actions to fully support the implementation of the general model to the specific situation. If the general model is to be successful, it must become an integral part of the system - planning, scheduling, assigning tasks, maintaining status, reporting progress, etc.

SELECTED BIBLIOGRAPHY

(1) Rothman, Stanley, and Charles Mosman. Computers and Society. Chicago: Science Research Associates, Inc., 1972.

(2) Barnard, Chester I. The Functions of the Executive. Cambridge, Mass.: Harvard University Press, 1938.

(3) Blumenthal, Sherman C. Management Information Systems: A Framework for Planning and Development. Englewood Cliffs, N. J.: Prentice-Hall, 1969.

(4) Benjamin, R. Control of Information System Development Cycle. New York: Wiley, 1971.

(5) Szweda, Ralph A. Information Processing Management. Princeton: Auerbach, 1972.

(6) Sanders, Donald H. Computers and Management. New York: McGraw-Hill, 1970.

(7) Albrecht, Leon K. Organization and Management of Information Processing Systems. New York: Macmillan, 1973.

(8) Martin, James. Programming Real-Time Computer Systems. Englewood Cliffs, N. J.: Prentice-Hall, 1965.

(9) Martin, James. Design of Real-Time Computer Systems. Englewood Cliffs, N. J.: Prentice-Hall, 1967.

(10) Flores, Ivan. Computer Programming. Englewood Cliffs, N. J.: Prentice-Hall, 1966.

(11) Flores, Ivan. Computer Software: Programming Systems for Digital Computers. Englewood Cliffs, N. J.: Prentice-Hall, 1965.

(12) Bocchino, William A. Management Information System Tools and Techniques. Englewood Cliffs, N. J.: Prentice-Hall, 1972.

(13) Ross, Joel E. Management by Information System. Englewood Cliffs, N. J.: Prentice-Hall, 1970.

(14) O'Brien, James J. Management Information Systems Concepts, Techniques, and Applications. New York: Van Nostrand Rheinhold, 1970.

(15) Kanter, Jerome. Management-Oriented Management Information Systems. Englewood Cliffs, N. J.: Prentice-Hall, 1972.

(16) Dearden, John F., Warren McFarlan, and William M. Zani. Managing Computer-Based Information Systems. Homewood, Ill.: Irwin, Inc., 1971.

(17) Avots, Ivars. "Making Project Management Work." Datamation, Vol. 19, No. 1 (Jan., 1973), 42-45.

(18) Management Planning Guide for a Manual of Data Processing Standards (C20-1670-1), IBM, 112 East Post Road, White Plains, N. Y.

(19) Modler, Joseph J., and Cecil R. Phillips, Project Management Techniques With CPM and PERT. New York: Van Nostrand Reinhold, 1970.

(20) MINIPERT Program Description and Operations Manual. Program Number 5734-XP3 (SH20-0995-0), IBM, 1133 Westchester Avenue, White Plains, N. Y.

(21) Ackoff, R. L. "Management Misinformation Systems." Management Science, Vol. 14, No. 4 (1967).

(22) Anthony, Robert N. Planning and Control Systems, A Framework for Analysis. Boston: Division of Research, Harvard Business School, 1965.

(23) Aron, Joel D. "Real-Time Systems in Perspective." IBM Systems Journal, Vol. 6, No. 1 (1967).

(24) Aron, Joel D. "Information Systems in Perspective." Information Systems Symposium, Washington, D. C., September 4-6, 1968.

(25) Awad, Elias M. Business Data Processing. Englewood Cliffs, N. J.: Prentice-Hall Inc., 1971.

(26) Bard, Y. "Performance Criteria and Measurement for a Time-Sharing System." IBM Systems Journal, Vol. 10, No. 3 (1971).

(27) Boehm, Barry W. "Software and Its Impact: A Quantitative Assessment." Datamation, Vol. 19, No. 5 (May, 1973).

(28) Boulden, J. B., and E. S. Buffa. "Corporate Models: On-Line, Real-Time Systems." Harvard Business Review, Vol. 48, No. 4 (July-August, 1970).

(29) Brandon, Dick H. Management Planning for Data Processing. Princeton, N. J.: Brandon Systems Press, 1970.

(30) Burlingame, J. F. "Information Technology and Decentralization." Harvard Business Review, Vol. 39, No. 6 (Nov.-Dec., 1961).

(31) Canning, R. G., and R. L. Sisson. The Management of Data Processing. New York: John Wiley and Sons, 1967.

(32) Carne, E. B. "Telecommunications: Its Impact on Business." Harvard Business Review, Vol. 50, No. 4 (July-August, 1972).

(33) Francis, Michael. "IBM's New System: Making It On a Network Basis," Finance, Vol. 90, No. 10 (November, 1972).

(34) Greunberger, F. Information Systems for Management. Englewood Cliffs, N. J.: Prentice-Hall, 1972.

(35) Head, Robert V. "Testing Real-Time Systems - Part I." Datamation, Vol. 10, No. 7 (July, 1964).

(36) Johnson, R. A., F. E. Kast, and J. E. Rosenzweig. The Theory and Management of Systems. New York: McGraw-Hill, 1967.

(37) McFarlan, F. Warren. "Management Audits of the EDP Department." Harvard Business Review, Vol. 51, No. 3 (May-June, 1973).

(38) Prince, T. R. Information Systems for Management Planning and Control. Homewood, Ill.: Irwin, 1970.

(39) Rubin, M. L. System Life Cycle Standards. Princeton, N. J.: Brandon Systems Press, 1970.

APPENDIX A

ACTIVITY DESCRIPTIONS FOR GENERAL MANAGEMENT INFORMATION SYSTEM MODEL

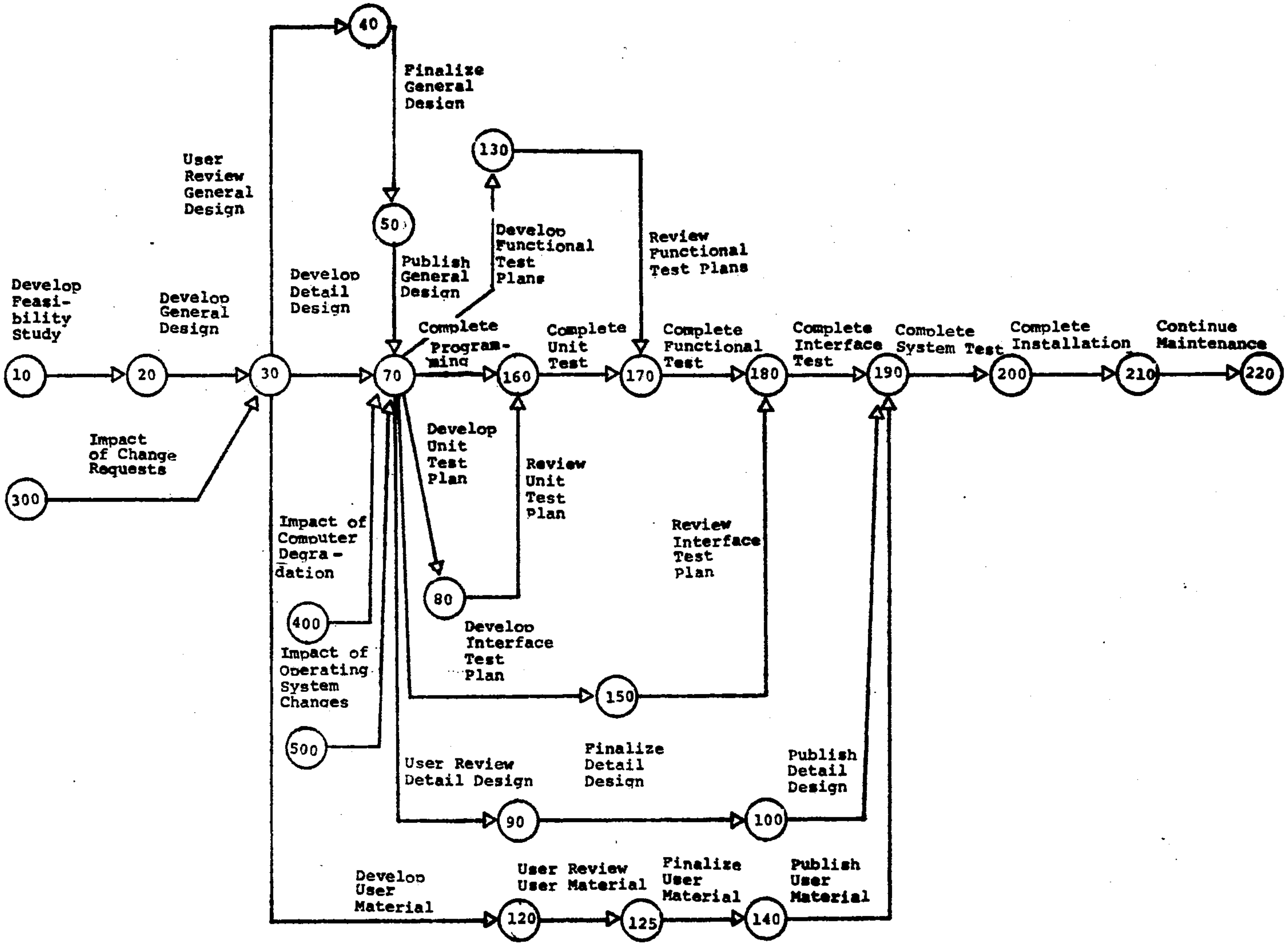


Figure 26. General Model of a Management Information System - Emphasis Testing

ACTIVITY NO: 10 - 20

ACTIVITY NAME: Develop Feasibility Study

DESCRIPTION: The feasibility study begins with a preliminary feasibility study. The full feasibility study considers the operational, technical, and economic aspects of the proposed system versus the existing system. Management must review final results and give GO or NO-GO decision.

INPUT: Need identified with very rough-cut benefits statement.

OUTPUT: Preliminary feasibility study is completed. It has developed reasons and objectives for full feasibility study, estimated cost for full feasibility study, recommended appropriate personnel for full feasibility study, and proposed a full feasibility schedule.

Statement of whether proposed system will operationally produce the required output within the response time requirements.

Statement on the technical considerations of software, hardware, and other support requirements.

Statement of all assumptions for future cross reference.

Statement of economic benefits with corresponding cost and saving identified for each benefit (with ROI and pay back period).

Statement of intangible benefits.

Management acceptance or rejection of feasibility study.

STAFFING: Top Management.

Users.

Key System Analyst.

Key Programmer.

ACTIVITY NO: 20 - 30

ACTIVITY NAME: Develop General Design

DESCRIPTION: Develop a general design which is at a very high level but yet identifies scope and significant features from initiation to completion of function.

INPUT: Top management approval of feasibility study.

OUTPUT: Very high statement of scope and significant features.

List of committed and approved functions inherent in design.

List of functions, which the user may want, which are not committed but are desirable and which will be considered later.

List of functions which will not be satisfied and probably will not be committed.

Statement of user control of system.

Flow chart of normal functions which are to be performed.

Use of exhibits, diagrams, and flow charts to make material meaningful.

General design has been reproduced and can be sent to user for review.

Statement of development and installation strategy.

List of contact names and telephone numbers of personnel responsible for various portions of general design.

STAFFING: Analysts expert in function.

Key programmer.

ACTIVITY NO: 30 - 40

ACTIVITY NAME: User Review General Design

DESCRIPTION: The user reviews the general design document for technical concurrence and readability. There may be many iterations between the user and the developer of the document before the user expresses his needs as mandatory, optional, or no changes. If at all possible, their should be complete communication so that the reply is mutually agreeable.

INPUT: Receipt of general design documentation (which contains the output checklist information from the activity entitled develop general design).

Contact names and telephone numbers for the appropriate personnel who are responsible for the various portions of general design.

OUTPUT: Return of general design document to the developer of the document with statement of change being mandatory, optional, or no changes.

User management concurs with the general design changes being termed mandatory, optional, or no changes.

STAFFING: User system analysts.

Key user programmer.

Appropriate user management (for statement of general design document requiring mandatory, optional, or no changes).

ACTIVITY NO: 30 - 70

ACTIVITY NAME: Develop Detail Design

DESCRIPTION: The detail design document describes in detail the sequence of events which the system must perform. It discusses the various application (groups of one or more programs) which make up the function. If there are changes made to the general design, by the user review or from other sources, these changes must be reflected in the detail design.

INPUT: General design documentation which has been sent to user.

Impact due to change requests.

Impact due to Operating System changes affecting interfaces.

OUTPUT: File records, file layouts, report formats, and editing and error requirements must be stated.

List of programs with number, name, function, purpose, and structure.

Detail design document which has developer management approval and is ready for user review.

List of all records which are updated, deleted, or referenced.

List of all output which could be messages, reports, or transactions.

Logic flow chart showing sequence of programming, the decisions and actions that take place along with the sequence of data to perform the various functions.

Logic flow chart narrative which describes the logic flow chart.

Program flow chart which describes the various programming steps.

Program flow chart narrative which describes the program flow chart.

STAFFING: System Analysts.

Designers.

Key Programmers.

Programmer/Analyst (if available is the best).

ACTIVITY NO: 30 - 120

ACTIVITY NAME: Develop User Material

DESCRIPTION: There must be a document which provides the user assistance in understanding the system. This includes such items as user procedures, reports and inquiries, operation procedures, education requirements, and dictionaries.

INPUT: General design documentation changes.

Detail design documentation changes.

OUTPUT: User procedures which describe how the users carry out their functions by defining job name and purpose, transactions name and purpose, interfaces transaction has with other transactions, hardware requirements by name and type, mode of operation, procedures for entering transactions, and output lists including error messages, action-required statements, etc.

Report and inquiry information including purpose, frequency, security requirements, restrictions, interfaces, entry procedures, hardware requirements, output messages, etc.

Operation instructions describing how to operate system under normal circumstances, how to recover and restart, how to answer questions when system is down.

Dictionary which defines various aspects of system.

Glossary which contains acronyms, abbreviations, definitions of terms.

Data element dictionary and user record dictionary.

Operator console messages defined.

Educational material and agenda should be stated.

STAFFING: System Analysts.

Key Programmers.

Key Designers.

<u>ACTIVITY NO:</u> 40 - 50

<u>ACTIVITY NAME:</u> Finalize General Design

<u>DESCRIPTION:</u> The developer of the general design reviews comments made by the user. The developer communicates, if necessary, with the user to completely understand the concerns. All mandatory changes must be resolved or escalated for resolution. Then, the general design is rewritten to reflect the user-developer agreements and is made ready for publication.

<u>INPUT:</u> General design document which has been reviewed by the user management and which contains statement of changes being mandatory, optional, or no changes.

<u>OUTPUT:</u> Developer management concerns with final general design document.

General design document has been written and typed to appropriate standards and is ready for publication activity.

<u>STAFFING:</u> Appropriate developer analysts.

Typists.

Technical writers.

Developer management (for final approval).

ACTIVITY NO: 50 - 70

ACTIVITY NAME: Publish General Design

DESCRIPTION: The general design document is reproduced and sent to an agreed upon distribution list.

INPUT: General design document which has been written and typed to appropriate standards and is ready for publication.

Distribution list which has been agreed to by developer and user.

OUTPUT: Copies of general design document ready to be sent to distribution list.

STAFFING: Reproduction personnel.

ACTIVITY NO: 70 - 80

ACTIVITY NAME: Develop Unit Test Plan

DESCRIPTIONS: Developer determines the combination of data input values which test all the functions of the programs. It should contain unexpected as well as expected input values to test the entire program (includes main logic and error routines). Also, it should be sequenced so that the smallest testable logic paths are tested in a rational sequence.

INPUT: Beginning of programming.

Detail design, general design, and user material documentation.

OUTPUT: Statement of system performance criteria and plan.

Unit test data defined which tests main and error logic.

Data bank is defined.

Input edits and audits must be stated.

Developer has listed all pre-defined results which are expected.

STAFFING: Programmers.

Analyst.

Programmer/Analyst (if available is best).

ACTIVITY NO: 70 - 90

ACTIVITY NAME: User Review Detail Design

DESCRIPTION: The user reviews the detail design document for technical concurrence and readability. There may be many iterations between the user and the developer of the document before the user expresses his needs as mandatory, optional, or no changes. If at all possible there should be complete communication so that the reply is mutually agreeable between the user and developer.

INPUT: Receipt of detail design documentation (which contains the output checklist information from the activity entitled develop detail design).

Contact name and telephone numbers for the appropriate personnel who are responsible for the various portions of detail design.

OUTPUT: Return of detail design document to the developer of the document with statement of change being mandatory, optional, or no changes.

User management concurs with the detail design changes being termed mandatory, optional, or no changes.

STAFFING: User system analysts.

Key user programmers.

Appropriate user management (for statement of detail design document requiring mandatory, optional, or no changes).

ACTIVITY NO: 70 - 110

ACTIVITY NAME: Develop System Test Plan

DESCRIPTION: The developers, testers, and users determine the combination of data input values which test all the functions of the entire system. It should be structured so that it is representative of the user's requirements, system configuration must also be representative. Also, a testing strategy and philosophy must be developed which is consistent with previous testing. Each test should build to a completely tested system.

INPUT: General design, detail design, user material documentation.

Input from unit, functional, and interface testing results.

OUTPUT: Determination of user's transaction volumes, timing characteristics, data bank record level, etc.

Performance criteria statement.

Data bank defined.

Pre-determined results listed.

Bill of material for all programs in release.

Statement of error messages.

Record levels defined.

STAFFING: Programmers (from developer, tester, and user).

Analysts (from developer, tester, and user).

ACTIVITY NO: 70 - 130

ACTIVITY NAME: Develop Functional Test Plan

DESCRIPTION: The developer determines the combination of data input values which test all the functions of the interfaces between various programs in a defined string. It should contain unexpected as well as expected input values to test the entire program (includes main logic and error routines). It should begin with the smallest testable path and keep increasing in a rational manner until the complete function is tested.

INPUT: Beginning of programming.

Unit testing experience as it becomes available.

Detail design, general design, and user material documentation.

OUTPUT: Statement of system performance criteria and plan.

Functional test data defined which tests main and error logic.

Records and data bank defined.

Input edits and audits must be stated.

Developer has listed all pre-defined results which are expected including transactions, messages, records altered and data elements updated.

STAFFING: Programmers.

Analysts.

Programmer/Analyst (if available).

ACTIVITY NO: 70 - 150

ACTIVITY NAME: Develop Interface Test Plan

DESCRIPTION: Developer determines the combination of data input values which test all the interfaces between the subsystems. It should contain unexpected as well as expected input values. The user should provide list of interfaces he expects to use. Interfaces are required when a particular function crosses two or more subsystems.

INPUT: Input from unit test results and functional test results which builds into interface test.

Detail design, general design, and user material documentation.

Input from unit test and functional test results.

OUTPUT: Statement of interface test performance criteria and plan.

Interface test data defined.

Data bank is defined.

All pre-defined results are listed.

STAFFING: Programmers (from developer and user).

Analyst (from developer and user).

Programmer/Analyst (if available is best).

ACTIVITY NO: 70 - 160

ACTIVITY NAME: Complete Programming

DESCRIPTION: This activity includes all work necessary for the coding, desk checking and compiling of the program until the first error free compile. The programming will be influenced by any changes made to the general design, detail design, or user material; therefore, communication between these areas must be maintained. Every effort should be made to attempt terminal debugging by using a real-time, on-line language.

INPUT: General design documentation and changes as they occur.

Detail design documentation and changes as they occur.

User material documentation and changes as they occur.

Impact of operating system changes as they occur.

Impact of change requests as they occur.

Impact of computer degradation as it occurs.

OUTPUT: Program which has completed its first error free compile (source to object without errors) to standards and documentation.

Management and technical committee review to insure that program has completed its first clean compile.

STAFFING: Programmer.

Key System Analyst (to answer design questions if they arise).

Programmer/Analyst (if available is the best).

ACTIVITY NO: 80 - 160

ACTIVITY NAME: Review Unit Test Plan

DESCRIPTION: The unit test plan should not only be reviewed by the developer, but also any area which will either be testing or installing the program. The test plan should be discussed to insure that all functions are accomplished in an agreed upon manner.

INPUT: Unit test plan which reflects developer's plans.

General design, detail design, and user material documentation.

OUTPUT: Reviewed unit test plan which contains management agreement from both the user and developer.

Unit test data completely defined and registered.

Data bank is completely defined and registered.

All pre-defined results have been listed.

Performers criteria have been agreed upon between user and developer.

STAFFING: Testing personnel (who will test program).

User personnel (who will be using program).

Programmers (from all affected areas).

Analysts (from all affected areas).

ACTIVITY NO: 90 - 100

ACTIVITY NAME: Finalize Detail Design

DESCRIPTION: The developer of the detail design reviews comments made by the user. The developer communicates, if necessary, with the user to completely understand the concerns. All mandatory changes must be resolved or escalated for resolution. Then, the detail design is rewritten to reflect the user-developer agreements and is made ready for publication.

INPUT: Detail design document which has been reviewed by the user management and which contains statement of changes being mandatory, optional, or no changes.

OUTPUT: Developer management concurs with final detail design document.

Detail design document has been written and typed to appropriate standards and is ready for publication activity.

STAFFING: Appropriate developer analysts.

Typists.

Technical writers.

Developer management (for final approval).

ACTIVITY NO: 100 - 190

ACTIVITY NAME: Publish Detail Design

DESCRIPTION: The detail design document is reproduced and sent to an agreed upon distribution list.

INPUT: Detail design document which has been written and typed to appropriate standards and is ready for publication.

Distribution list which has been agreed to by developer and user.

OUTPUT: Copies of detail design document ready to be sent to distribution list.

STAFFING: Reproduction personnel.

ACTIVITY NO: 110 - 190

ACTIVITY NAME: Review System Test Plan

DESCRIPTION: All areas should review the system test plan. There should be particular emphasis that the plan represents the user's characteristics. If there are any disagreements to the plans content, the user's recommendation should be followed. Agreement on a consistent, non-redundant testing philosophy.

INPUT: System test plan.

General design, detail design, and user material documentation statement of testing philosophy.

OUTPUT: Statement of data bank record levels.

Pre-defined results have been stated.

Reviewed system test plan which contains management agreement from both the user and developer.

System test data completely defined and registered.

Performance criteria has been agreed upon and stated.

User transaction mix, volumes, and timing characteristics.

Statement of expected report formats, user messages, operator messages and responses.

STAFFING: Programmers (from all affected areas).

Analysts (from all affected areas).

ACTIVITY NO: 120-125

ACTIVITY NAME: User Review User Material

DESCRIPTION: The user reviews the user material document for technical concurrence and readability. There may be many iterations between the user and the developer of the document before the user expresses his needs as mandatory, optional, or no changes. If at all possible there should be complete communication so that the reply is mutually agreeable between the user and developer.

INPUT: Receipt of user material documentation (which contains the output checklist information from the activity called develop user material).

Contact names and telephone numbers for the appropriate personnel who are responsible for the various portions of the user material.

OUTPUT: Return of user material document to the developer of the document with statement of change being mandatory or optional.

User management concurs with the user material changes being termed mandatory, optional, or no changes.

STAFFING: User system analysts.

Key user programmer.

Appropriate user management (for statement of the user material document requiring mandatory, optional, or no changes).

ACTIVITY NO: 125 - 140

ACTIVITY NAME: Finalize User Material

DESCRIPTION: The developer of the user material reviews comments made by the user. The developer communicates, if necessary, with the user to completely understand the concerns. All mandatory changes must be resolved or escalated for resolution. Then, the general design is rewritten to reflect the user-developer agreements and is made ready for publication.

INPUT: User material documentation which has been reviewed by the user management and which contains statement of changes being mandatory, optional, or no changes.

OUTPUT: Developer management concurs with final user material document.

User material document has been written and typed to appropriate standards and is ready for publication activity.

STAFFING: Appropriate developer analysts.

Typists.

Technical writers.

Developer management (for final approval).

ACTIVITY NO: 130 - 170

ACTIVITY NAME: Review Functional Test Plan

DESCRIPTION: The functional test plan should not only be reviewed by the developer, but also any area which will either be testing or installing the program. The test plan should be reviewed so that all areas have contributed to the final functional test plan.

INPUT: Functional test plan which reflects developers plans.

General design, detail design, and user material documentation.

OUTPUT: Reviewed functional test plan which contains management agreement from both the user and developer.

Functional test data completely defined and registered.

Data bank is completely defined and registered.

All pre-defined results have been agreed upon between user and developer.

STAFFING: Testing personnel (who will test program).

User personnel (who will be using program).

Programmers (from all affected areas).

Analysts (from all affected areas).

ACTIVITY NO: 140 - 190

ACTIVITY NAME: Publish User Material

DESCRIPTION: The user material document is reproduced and sent to an agreed upon distribution list.

INPUT: User material document which has been written and typed to appropriate standards and is ready for publication.

Distribution list which has been agreed to by developer and user.

OUTPUT: Copies of user material document ready to be sent to distribution list.

STAFFING: Reproduction personnel.

ACTIVITY NO: 150 - 180

ACTIVITY NAME: Review Interface Test Plan

DESCRIPTION: The interface test plan should not only be reviewed by the developer, but also any area which will either be testing or installing the program. The test plan should be discussed to insure that all functions are accomplished in an agreed upon manner.

INPUT: Interface test plan.

General design, detail design, and user material documentation.

OUTPUT: Reviewed interface test plan which contains management agreement from both the user and developer.

Interface test data completely defined and registered.

Interface data bank is completely defined and registered.

All pre-defined results have been listed.

Performance criteria have been agreed upon between user and developer for interface test.

STAFFING: Testing personnel (who will test program).

User personnel (who will be using program).

Programmers (from all affected areas).

Analysts (from all affected areas).

ACTIVITY NO: 160 - 170

ACTIVITY NAME: Complete Unit Test

DESCRIPTION: The test data, which reflects the unit test plan, is run which verifies that the program runs to predicted completion. It is the least expensive level of testing which is performed. As in the programming stage, the testing should be done via terminal and should be done on the smallest possible segments.

INPUT: General design, detail design, and user documentation.

Error free compilation.

Unit test plan developed and reviewed.

Unit test data and data bank is registered.

OUTPUT: All error routines have been executed.

Performance criteria has been measured.

Insurance that report formats, user message format, operator messages and responses are satisfactory.

Unit tested programs that have passed the unit test plans and agree with documentation.

Management - technical concurrence that unit test has been completed.

Insure that the highest possible percentage of code has been tested in each program (run a test program if required).

STAFFING: Programmers.

System Analyst (as required).

Programmer/Analyst (if available is best).

ACTIVITY NO: 170 - 180

ACTIVITY NAME: Complete Functional Test

DESCRIPTION: The functional test consists of testing inter-subsystem interfaces. The various programs within a subsystem are interfaced together (strings) according to pre-defined functions defined in the functional test plan. A string has a beginning point and a terminal point. After the string has satisfactorily (according to functional test plan criteria) processed the input, it is considered to have completed functional test.

INPUT: Completion of Unit Test (see output checklist for activity entitled complete unit test).

Completion of functional test plan review (see output checklist for activity entitled review functional test plan).

Functional test data bank established.

Bill of material for programs entering functional test.

General design, detail design, and user material documentation.

OUTPUT: Program to program interfaces have been executed and verified for each string according to functional test plan.

Additional test conditions (from unit test) have been executed to insure adequacy of test across programs.

Documentation has been verified (understandable and useable) that it conforms to programs.

Management - technical concurrence that functional test has been completed.

STAFFING: Programmers.

Key System Analyst.

Programmer/Analyst (if available it is best).

ACTIVITY NO: 180 - 190

ACTIVITY NAME: Complete Interface Test

DESCRIPTION: The interfaces between the subsystems are tested to insure that the programs run to predicted completion according to interface test plan. The control over the progress of the interface testing should be at the string level.

INPUT:

Bill of material for all programs entering interface test.

Complete functional test.

General design, detail design, and user documentation.

Developer management concurs that programs are ready to enter interface test.

Interface test plan developed and reviewed.

Interface test data, data bank, and records are developed and registered.

Cross reference list between strings, load modules, test data, records, etc.

OUTPUT:

Subsystem to subsystem interfaces have been executed and verified for each string according to interface test plan and documentation.

All error routines have been executed.

Performance criteria has been measured.

Management - technical concurrence that interface test has been completed.

Insurance that report formats, user message format, operator messages and responses are satisfactory.

STAFFING:

Programmer.

Analysts (from developer and user).

Programmer/Analyst.

ACTIVITY NO: 190 - 200

ACTIVITY NAME: Complete System Test

DESCRIPTION: System test is an independent test and is critical to the success of the program. It includes all the work necessary to test the subsystems and their inherent programs under realistic operating conditions. It is a real-time, on-line test under operating conditions which simulate the installing locations transaction volumes, timing characteristics, data bank record levels, etc. The tests should insure that the documentation supports the actual operation and also, that the performance criteria are met. All testing should be completed in this test except for any testing which it is technically not possible to accomplish (bridges, and other items which are unique to one user).

INPUT: System test plan which has been reviewed by user and developer.

All documentation (general design, detail design, user material) to support system operation.

All testing has been completed except for system test.

Developer management concurs that programs are ready to enter system test.

Bill of material for all programs entering system test.

OUTPUT: List of any programs which are not functioning to specification and fix date.

Insure that documentation agrees with system test results.

Validate fall back and recovery procedures.

Validate that system meets performance criteria.

Insure that system operates in full volume environment.

Insure that documentation supports operation.

All error routines have been executed.

Insurance that report formats, user message formats, operator messages and responses are satisfactory.

Management - technical concurrence that system test has been completed.

Insure that the highest possible percentage of code has been tested in system (run test program if required).

STAFFING: Programmers (from user location and developer's only when required).

Analysts (from user's location and developer's only when required).

Programmer/Analyst (if available is best).

ACTIVITY NO: 200 - 210

ACTIVITY NAME: Complete Installation

DESCRIPTION: This activity is where the results of all the previous work should be realized. The programs which have completed system test are installed at the user's location. There is generally a considerable amount of work at the user's location for education of affected personnel, bridge programs, compatible procedures with new programs, etc. The new programs will not be cut-over immediately, but there must be a planned, structured, parallel test which have defined cut-over criteria. Cut-over to the new programs will be accomplished after satisfactory testing, parallel-run criteria have been met and other planned activities have been completed. There should be an expected hump in the installation resources required when the new programs are cut-over. This is to correct the problems which could not be found earlier.

INPUT: The physical installation site must be made ready for installation.

Programs which have completed all the criteria in the output checklist of the activity entitled complete system test.

Insurance that previous testing had been supported by users so that new programs are known.

General design, detail design, and user material documentation which fully supports the system tested programs.

List of all off-specification programs from system test activities.

Bill of material of all programs, records, etc.

Statement of support requirements from developers and testers.

Detailed conversion schedules must be developed and used to insure progress.

Study to determine the optimum method of installing the new programs (includes schedules, logical installation sequence).

Organization philosophy of user location is stated.

System operating procedures have been defined at user location.

File initialization and file conversion programs are ready and have been tested.

Determination of how the hardware requirements are to be planned.

Recovery and reset procedures are user defined.

Data bank integrity procedures have been developed.

The programs have been cut-over and are driving the user location and have been signed off-on by the user management.

STAFFING: Programmers (from user location but with developer support).

Analysts (from user location but with developer support).

ACTIVITY NO: 210 - 220

ACTIVITY NAME: Continue Maintenance

DESCRIPTION: The maintenance of the programs is an on-going activity. The programs after being installed do not continually operate in a static environment. Software is upgraded, procedures are changed, performance improvements are desired, etc. All of these changes must be considered and the system must be maintained accordingly. The changes must be made in consistent manner or else havoc will result. Each change must have a formal request for change, statement of need, cost analysis, and a proposed schedule which is reviewed and concurred with by appropriate management. If the programs are used at more than one location, special emphasis must be placed on insuring that the fixes are tested at one central location and then distributed to all other areas.

INPUT: Procedure which must be followed when changing programs which includes formal request for change, statement of need, cost analysis, schedules, progress control.

Technical procedures for knowing how to make change, where change should be tested, and how to control the various program levels across all affected areas.

OUTPUT: Programs are continually up-to-date.

STAFFING: Programmers (at user, testing, and developer locations).

Analysts (at user, testing, and developer locations).

ACTIVITY NO: 300 - 30

ACTIVITY NAME: Impact of Change Requests

DESCRIPTION: Change requests are statements of requested change to the programs due to performance improvements, maintenance requirements, or operating procedures. There must be a procedure which is followed by all affected areas (developers, testers, and users). The key concern with change requests is usually the communication of appropriate information to all areas. Each change request is unique and must be sized on a one-for-one basis. Management must be constantly aware of the situation so that appropriate schedules, progress, and commitments can be made.

INPUT: Change request procedure which is commonly agreed upon and used by the developer, tester, and user (includes forms, flow of information, checkpoint meetings, etc.).

One central repository of key information concerning all change requests.

OUTPUT: Impact of change request on management information system in terms of resources (dollars) and schedule changes.

STAFFING: Affects all areas (developers, testers, and users).

ACTIVITY NO: 400 - 70

ACTIVITY NAME: Impact of Computer Degradation

DESCRIPTION: The changes caused by computer degradation are considered by many to be the most critical factor in contributing to schedule slippages. The degradation can be either a reduced turnaround time or actually having the computer become inoperable. It is a constantly fluctuating activity which is difficult to size and understand. However, there are some basic problems which must be overcome. There must be common definition of key values. Then, there must be continual communication between all affected areas to insure that the problems are being corrected.

INPUT: Procedure for reporting status on computer turnaround and down-time hours.

Communication meetings to insure that critical problems are being addressed.

Establishment of a priority list for insuring that the most critical items are being addressed across all affected areas.

OUTPUT: Impact of computer degradation in terms of resources (dollars) and schedule changes.

STAFFING: All affected areas (developers, testers, and users).

ACTIVITY NO: 500 - 70

ACTIVITY NAME: Impact of Operating System Changes

DESCRIPTION: There usually are changes made to the operating system (O/S) due to maintenance changes, performance improvements, and enhancement commitments. The changes can be either transparent (application programs are affected but no changes required) or can cause changes to be madeto the application programs. Often the changes are termed transparent but when the change is installed, there are actually changes required to the application programs. There must be continual communication to insure that all areas are aware the change status and planned changes.

ment

INPUT: Statement of Operating System (O/S) change and related information.

Communication mechanism to insure that the affected areas know when, where, why, how the software change will be made.

OUTPUT: Impact of operating system changes in terms of resources (dollars) and schedule changes.

STAFFING: All affected areas (developers, testers, and users).

APPENDIX B

SELECTED OUTPUT REPORTS FROM GENERAL MANAGEMENT INFORMATION SYSTEM MODEL

PROJECT: M.I.S. GENERAL MODEL PRINTED 04/05/73 AT 15.16.20

TIMENOW DATE USED:

SORTED BY EARLY START

PRED	SUCC	DUR	ES	EF	LS	LF	SLACK	SCHD	DESCRIPTION
10	20	10.00	04/02/73	04/16/73	04/02/73	04/16/73	0	04/02/73	DEVELOP FEASIBILITY STUDY
20	30	12.00	04/16/73	05/02/73	04/16/73	05/02/73	0		DEVELOP GENERAL DESIGN
30	40	10.00	05/02/73	05/16/73	05/09/73	05/23/73	5.00		USER REVIEW GENERAL DESIGN
30	70	35.00	05/02/73	06/20/73	05/02/73	06/20/73	0		DEVELOP DETAIL DESIGN
30	120	20.00	05/02/73	05/30/73	08/13/73	09/10/73	73.00		DEVELOP USER MATERIAL
300	30	0	05/02/73	05/02/73	05/02/73	05/02/73	0	05/02/73	IMPACT OF CHANGE REQUESTS
400	70	0	05/15/73	05/15/73	06/20/73	06/20/73	26.00	05/15/73	IMPACT OF COMPUTER DEGRADATION
40	50	10.00	05/16/73	05/30/73	05/23/73	06/06/73	5.00		FINALIZE GENERAL DESIGN
50	70	10.00	05/30/73	06/13/73	06/06/73	06/20/73	5.00		PUBLISH GENERAL DESIGN
120	125	10.00	05/30/73	06/13/73	09/10/73	09/24/73	73.00		USER REVIEW USER MATERIAL
500	70	0	06/04/73	06/04/73	06/20/73	06/20/73	12.00	06/04/73	IMPACT OF OPERATING SYSTEM CHANGES
125	140	10.00	06/13/73	06/27/73	09/24/73	10/08/73	73.00		FINALIZE USER MATERIAL
70	80	20.00	06/20/73	07/18/73	07/09/73	08/06/73	13.00		DEVELOP UNIT TEST PLAN
70	90	10.00	06/20/73	07/04/73	09/10/73	09/24/73	58.00		USER REVIEW DETAIL DESIGN
70	110	70.00	06/20/73	09/26/73	07/02/73	10/08/73	8.00		DEVELOP SYSTEM TEST PLAN
70	130	30.00	06/20/73	08/01/73	07/16/73	08/27/73	18.00		DEVELOP FUNCTIONAL TEST PLAN
70	150	40.00	06/20/73	08/15/73	07/23/73	09/17/73	23.00		DEVELOP INTERFACE TEST PLAN
70	160	43.00	06/20/73	08/20/73	06/20/73	08/20/73	0		COMPLETE PROGRAMMING
140	190	10.00	06/27/73	07/11/73	10/08/73	10/22/73	73.00		PUBLISH USER MATERIAL
90	100	10.00	07/04/73	07/18/73	09/24/73	10/08/73	58.00		FINALIZE DETAIL DESIGN
80	160	10.00	07/18/73	08/01/73	08/06/73	08/20/73	13.00		REVIEW UNIT TEST PLAN
100	190	10.00	07/18/73	08/01/73	10/08/73	10/22/73	58.00		PUBLISH DETAIL DESIGN
130	170	10.00	08/01/73	08/15/73	08/27/73	09/10/73	18.00		REVIEW FUNCTIONAL TEST PLAN
150	180	10.00	08/15/73	08/29/73	09/17/73	10/01/73	23.00		REVIEW INTERFACE TEST PLAN
160	170	15.00	08/20/73	09/10/73	08/20/73	09/10/73	0		COMPLETE UNIT TEST
170	180	15.00	09/10/73	10/01/73	09/10/73	10/01/73	0		COMPLETE FUNCTIONAL TEST
110	190	10.00	09/26/73	10/10/73	10/08/73	10/22/73	8.00		REVIEW SYSTEM TEST PLAN
180	190	15.00	10/01/73	10/22/73	10/01/73	10/22/73	0		COMPLETE INTERFACE TEST
190	200	50.00	10/22/73	12/31/73	10/22/73	12/31/73	0		COMPLETE SYSTEM TEST
200	210	28.00	12/31/73	02/07/74	12/31/73	02/07/74	0		COMPLETE INSTALLATION
210	220	0	02/07/74	02/07/74	02/07/74	02/07/74	0		CONTINUE MAINTENANCE

PROJECT: M.I.S. GENERAL MODEL PRINTED 04/05/73 AT 15.22.33

TIMENOW DATE USED:

SORTED BY LATE START

PRED	SUCC	DUR	ES	EF	LS	LF	SLACK	SCHD	DESCRIPTION
10	20	10.00	04/02/73	04/16/73	04/02/73	04/16/73	0	04/02/73	DEVELOP FEASIBILITY STUDY
20	30	12.00	04/16/73	05/02/73	04/16/73	05/02/73	0		DEVELOP GENERAL DESIGN
30	70	35.00	05/02/73	06/20/73	05/02/73	06/20/73	0		DEVELOP DETAIL DESIGN
300	30	0	05/02/73	05/02/73	05/02/73	05/02/73	0	05/02/73	IMPACT OF CHANGE REQUESTS
30	40	10.00	05/02/73	05/16/73	05/09/73	05/23/73	5.00		USER REVIEW GENERAL DESIGN
40	50	10.00	05/16/73	05/30/73	05/23/73	06/06/73	5.00		FINALIZE GENERAL DESIGN
50	70	10.00	05/30/73	06/13/73	06/06/73	06/20/73	5.00		PUBLISH GENERAL DESIGN
70	160	43.00	06/20/73	08/20/73	06/20/73	08/20/73	0		COMPLETE PROGRAMMING
400	70	0	05/15/73	05/15/73	06/20/73	06/20/73	26.00	05/15/73	IMPACT OF COMPUTER DEGRADATION
500	70	0	06/04/73	06/04/73	06/20/73	06/20/73	12.00	06/04/73	IMPACT OF OPERATING SYSTEM CHANGES
70	110	70.00	06/20/73	09/26/73	07/02/73	10/08/73	8.00		DEVELOP SYSTEM TEST PLAN
70	80	20.00	06/20/73	07/18/73	07/09/73	08/06/73	13.00		DEVELOP UNIT TEST PLAN
70	130	30.00	06/20/73	08/01/73	07/16/73	08/27/73	18.00		DEVELOP FUNCTIONAL TEST PLAN
70	150	40.00	06/20/73	08/15/73	07/23/73	09/17/73	23.00		DEVELOP INTERFACE TEST PLAN
80	160	10.00	07/18/73	08/01/73	08/06/73	08/20/73	13.00		REVIEW UNIT TEST PLAN
30	120	20.00	05/02/73	05/30/73	08/13/73	09/10/73	73.00		DEVELOP USER MATERIAL
160	170	15.00	08/20/73	09/10/73	08/20/73	09/10/73	0		COMPLETE UNIT TEST
130	170	10.00	08/01/73	08/15/73	08/27/73	09/10/73	18.00		REVIEW FUNCTIONAL TEST PLAN
70	90	10.00	06/20/73	07/04/73	09/10/73	09/24/73	58.00		USER REVIEW DETAIL DESIGN
170	180	15.00	09/10/73	10/01/73	09/10/73	10/01/73	0		COMPLETE FUNCTIONAL TEST
120	125	10.00	05/30/73	06/13/73	09/10/73	09/24/73	73.00		USER REVIEW USER MATERIAL
150	180	10.00	08/15/73	08/29/73	09/17/73	10/01/73	23.00		REVIEW INTERFACE TEST PLAN
90	100	10.00	07/04/73	07/18/73	09/24/73	10/08/73	58.00		FINALIZE DETAIL DESIGN
125	140	10.00	06/13/73	06/27/73	09/24/73	10/08/73	73.00		FINALIZE USER MATERIAL
180	190	15.00	10/01/73	10/22/73	10/01/73	10/22/73	0		COMPLETE INTERFACE TEST
100	190	10.00	07/18/73	08/01/73	10/08/73	10/22/73	58.00		PUBLISH DETAIL DESIGN
110	190	10.00	09/26/73	10/10/73	10/08/73	10/22/73	8.00		REVIEW SYSTEM TEST PLAN
140	190	10.00	06/27/73	07/11/73	10/08/73	10/22/73	73.00		PUBLISH USER MATERIAL
190	200	50.00	10/22/73	12/31/73	10/22/73	12/31/73	0		COMPLETE SYSTEM TEST
200	210	28.00	12/31/73	02/07/74	12/31/73	02/07/74	0		COMPLETE INSTALLATION
210	220	0	02/07/74	02/07/74	02/07/74	02/07/74	0		CONTINUE MAINTENANCE

PROJECT: M.I.S. GENERAL MODEL PRINTED 04/05/73 AT 15.28.17

TIMENOW DATE USED:

SORTED BY SLACK

PRED	SUCC	DUR	ES	EF	LS	LF	SLACK	SCHD	DESCRIPTION
20	30	12.00	04/16/73	05/02/73	04/16/73	05/02/73	0		DEVELOP GENERAL DESIGN
30	70	35.00	05/02/73	06/20/73	05/02/73	06/20/73	0		DEVELOP DETAIL DESIGN
70	160	43.00	06/20/73	08/20/73	06/20/73	08/20/73	0		COMPLETE PROGRAMMING
160	170	15.00	08/20/73	09/10/73	08/20/73	09/10/73	0		COMPLETE UNIT TEST
170	180	15.00	09/10/73	10/01/73	09/10/73	10/01/73	0		COMPLETE FUNCTIONAL TEST
180	190	15.00	10/01/73	10/22/73	10/01/73	10/22/73	0		COMPLETE INTERFACE TEST
190	200	50.00	10/22/73	12/31/73	10/22/73	12/31/73	0		COMPLETE SYSTEM TEST
200	210	28.00	12/31/73	02/07/74	12/31/73	02/07/74	0		COMPLETE INSTALLATION
210	220	0	02/07/74	02/07/74	02/07/74	02/07/74	0		CONTINUE MAINTENANCE
300	30	0	05/02/73	05/02/73	05/02/73	05/02/73	0	05/02/73	IMPACT OF CHANGE REQUESTS
10	20	10.00	04/02/73	04/16/73	04/02/73	04/16/73	0	04/02/73	DEVELOP FEASIBILITY STUDY
30	40	10.00	05/02/73	05/16/73	05/09/73	05/23/73	5.00		USER REVIEW GENERAL DESIGN
40	50	10.00	05/16/73	05/30/73	05/23/73	06/06/73	5.00		FINALIZE GENERAL DESIGN
50	70	10.00	05/30/73	06/13/73	06/06/73	06/20/73	5.00		PUBLISH GENERAL DESIGN
70	110	70.00	06/20/73	09/26/73	07/02/73	10/08/73	8.00		DEVELOP SYSTEM TEST PLAN
110	190	10.00	09/26/73	10/10/73	10/08/73	10/22/73	8.00		REVIEW SYSTEM TEST PLAN
500	70	0	06/04/73	06/04/73	06/20/73	06/20/73	12.00	06/04/73	IMPACT OF OPERATING SYSTEM CHANGES
70	80	20.00	06/20/73	07/18/73	07/09/73	08/06/73	13.00		DEVELOP UNIT TEST PLAN
80	160	10.00	07/18/73	08/01/73	08/06/73	08/20/73	13.00		REVIEW UNIT TEST PLAN
70	130	30.00	06/20/73	08/01/73	07/16/73	08/27/73	18.00		DEVELOP FUNCTIONAL TEST PLAN
130	170	10.00	08/01/73	08/15/73	08/27/73	09/10/73	18.00		REVIEW FUNCTIONAL TEST PLAN
70	150	40.00	06/20/73	08/15/73	07/23/73	09/17/73	23.00		DEVELOP INTERFACE TEST PLAN
150	180	10.00	08/15/73	08/29/73	09/17/73	10/01/73	23.00		REVIEW INTERFACE TEST PLAN
400	70	0	05/15/73	05/15/73	06/20/73	06/20/73	26.00	05/15/73	IMPACT OF COMPUTER DEGRADATION
70	90	10.00	06/20/73	07/04/73	09/10/73	09/24/73	58.00		USER REVIEW DETAIL DESIGN
90	100	10.00	07/04/73	07/18/73	09/24/73	10/08/73	58.00		FINALIZE DETAIL DESIGN
100	190	10.00	07/18/73	08/01/73	10/08/73	10/22/73	58.00		PUBLISH DETAIL DESIGN
30	120	20.00	05/02/73	05/30/73	08/13/73	09/10/73	73.00		DEVELOP USER MATERIAL
140	190	10.00	06/27/73	07/11/73	10/08/73	10/22/73	73.00		PUBLISH USER MATERIAL
125	140	10.00	06/13/73	06/27/73	09/24/73	10/08/73	73.00		FINALIZE USER MATERIAL
120	125	10.00	05/30/73	06/13/73	09/10/73	09/24/73	73.00		USER REVIEW USER MATERIAL

PROJECT: M.I.S. GENERAL MODEL PRINTED 04/05/73 AT 15.39.53

TIMENOW DATE USED:

SORTED BY PRED-SUCC

PRED	SUCC	DUR	ES	EF	LS	LF	SLACK	SCHD	DESCRIPTION
10	20	10.00	04/02/73	04/16/73	04/02/73	04/16/73	0	04/02/73	DEVELOP FEASIBILITY STUDY
20	30	12.00	04/16/73	05/02/73	04/16/73	05/02/73	0		DEVELOP GENERAL DESIGN
30	40	10.00	05/02/73	05/16/73	05/09/73	05/23/73	5.00		USER REVIEW GENERAL DESIGN
30	70	35.00	05/02/73	06/20/73	05/02/73	06/20/73	0		DEVELOP DETAIL DESIGN
30	120	20.00	05/02/73	05/30/73	08/13/73	09/10/73	73.00		DEVELOP USER MATERIAL
40	50	10.00	05/16/73	05/30/73	05/23/73	06/06/73	5.00		FINALIZE GENERAL DESIGN
50	70	10.00	05/30/73	06/13/73	06/06/73	06/20/73	5.00		PUBLISH GENERAL DESIGN
70	80	20.00	06/20/73	07/18/73	07/09/73	08/06/73	13.00		DEVELOP UNIT TEST PLAN
70	90	10.00	06/20/73	07/04/73	09/10/73	09/24/73	58.00		USER REVIEW DETAIL DESIGN
70	110	70.00	06/20/73	09/26/73	07/02/73	10/08/73	8.00		DEVELOP SYSTEM TEST PLAN
70	130	30.00	06/20/73	08/01/73	07/16/73	08/27/73	18.00		DEVELOP FUNCTIONAL TEST PLAN
70	150	40.00	06/20/73	08/15/73	07/23/73	09/17/73	23.00		DEVELOP INTERFACE TEST PLAN
70	160	43.00	06/20/73	08/20/73	06/20/73	08/20/73	0		COMPLETE PROGRAMMING
80	160	10.00	07/18/73	08/01/73	08/06/73	08/20/73	13.00		REVIEW UNIT TEST PLAN
90	100	10.00	07/04/73	07/18/73	09/24/73	10/08/73	58.00		FINALIZE DETAIL DESIGN
100	190	10.00	07/18/73	08/01/73	10/08/73	10/22/73	58.00		PUBLISH DETAIL DESIGN
110	190	10.00	09/26/73	10/10/73	10/08/73	10/22/73	8.00		REVIEW SYSTEM TEST PLAN
120	125	10.00	05/30/73	06/13/73	09/10/73	09/24/73	73.00		USER REVIEW USER MATERIAL
125	140	10.00	06/13/73	06/27/73	09/24/73	10/08/73	73.00		FINALIZE USER MATERIAL
130	170	10.00	08/01/73	08/15/73	08/27/73	09/10/73	18.00		REVIEW FUNCTIONAL TEST PLAN
140	190	10.00	06/27/73	07/11/73	10/08/73	10/22/73	73.00		PUBLISH USER MATERIAL
150	180	10.00	08/15/73	08/29/73	09/17/73	10/01/73	23.00		REVIEW INTERFACE TEST PLAN
160	170	15.00	08/20/73	09/10/73	08/20/73	09/10/73	0		COMPLETE UNIT TEST
170	180	15.00	09/10/73	10/01/73	09/10/73	10/01/73	0		COMPLETE FUNCTIONAL TEST
180	190	15.00	10/01/73	10/22/73	10/01/73	10/22/73	0		COMPLETE INTERFACE TEST
190	200	50.00	10/22/73	12/31/73	10/22/73	12/31/73	0		COMPLETE SYSTEM TEST
200	210	28.00	12/31/73	02/07/74	12/31/73	02/07/74	0		COMPLETE INSTALLATION
210	220	0	02/07/74	02/07/74	02/07/74	02/07/74	0		CONTINUE MAINTENANCE
300	30	0	05/02/73	05/02/73	05/02/73	05/02/73	0	05/02/73	IMPACT OF CHANGE REQUESTS
400	70	0	05/15/73	05/15/73	06/20/73	06/20/73	26.00	05/15/73	IMPACT OF COMPUTER DEGRADATION
500	70	0	06/04/73	06/04/73	06/20/73	06/20/73	12.00	06/04/73	IMPACT OF OPERATING SYSTEM CHANGES

```
PROJECT:     M.I.S.  GENERAL MODEL                PRINTED 04/05/73 AT 15.47.13

TIMENOW DATE USED:

SORTED BY EARLY START

O-SPAN OF EARLY START TO EARLY FINISH
*-SPAN OF LATE START TO LATE FINISH
●-COMPLETED ACTIVITY EARLY START TO EARLY FINISH

-----------------------------------------------------------------------------------------------------------------------
| |                     |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                     | |    1972    |     10     :      20  19:73   30     :      40     |    1974    |    1975    | |
-----------------------------------------------------------------------------------------------------------------------
| | 10      20          | |            |            :●●●         :            :            |            |            | |
| |DEVELOP FEASIBILITY STUDY           |            :            :            :            |            |            | |
| | 20      30          | |            |            :  ●●●       :            :            |            |            | |
| |DEVELOP GENERAL DESIGN              |            :            :            :            |            |            | |
| | 30      40          | |            |            :    O●●*    :            :            |            |            | |
| |USER REVIEW GENERAL DESIGN          |            :            :            :            |            |            | |
| | 30      70          | |            |            :    ●●●●●●● :            :            |            |            | |
| |DEVELOP DETAIL DESIGN               |            :            :            :            |            |            | |
| | 30      120         | |            |            :    OOOO****:**********  :            |            |            | |
| |DEVELOP USER MATERIAL               |            :            :            :            |            |            | |
| | 300     30          | |            |            :    ●       :            :            |            |            | |
| |IMPACT OF CHANGE REQUESTS           |            :            :            :            |            |            | |
| | 400     70          | |            |            :      O**** :            :            |            |            | |
| |IMPACT OF COMPUTER DEGRADATION      |            :            :            :            |            |            | |
| | 40      50          | |            |            :      O●*   :            :            |            |            | |
| |FINALIZE GENERAL DESIGN             |            :            :            :            |            |            | |
| | 50      70          | |            |            :       O●●* :            :            |            |            | |
| |PUBLISH GENERAL DESIGN              |            :            :            :            |            |            | |
| | 120     125         | |            |            :       OOO**:************:            |            |            | |
| |USER REVIEW USER MATERIAL           |            :            :            :            |            |            | |
| | 500     70          | |            |            :        O** :            :            |            |            | |
| |IMPACT OF OPERATING SYSTEM CHANGES  |            :            :            :            |            |            | |
| | 125     140         | |            |            :         OOO:************:**          |            |            | |
| |FINALIZE USER MATERIAL              |            :            :            :            |            |            | |
| | 70      80          | |            |            :          OO:O●●**      :            |            |            | |
| |DEVELOP UNIT TEST PLAN              |            :            :            :            |            |            | |
| | 70      90          | |            |            :          OO:O***********:            |            |            | |
| |USER REVIEW DETAIL DESIGN           |            :            :            :            |            |            | |
| | 70      110         | |            |            :          OO:●●●●●●●●●●●●:**          |            |            | |
| |DEVELOP SYSTEM TEST PLAN            |            :            :            :            |            |            | |
| | 70      130         | |            |            :          OO:OO●●●***   :            |            |            | |
| |DEVELOP FUNCTIONAL TEST PLAN        |            :            :            :            |            |            | |
| | 70      150         | |            |            :          OO:OOO●●●●**** :            |            |            | |
| |DEVELOP INTERFACE TEST PLAN         |            :            :            :            |            |            | |
| | 70      160         | |            |            :          ●●:●●●●●●●    :            |            |            | |
| |COMPLETE PROGRAMMING|               |            :            :            :            |            |            | |
| | 140     190         | |            |            :           O:OO**********:****        |            |            | |
| |PUBLISH USER MATERIAL               |            :            :            :            |            |            | |
| | 90      100         | |            |            :            :OOO*********:**          |            |            | |
| |FINALIZE DETAIL DESIGN              |            :            :            :            |            |            | |
| | 80      160         | |            |            :            :  OO●**     :            |            |            | |
| |REVIEW UNIT TEST PLAN               |            :            :            :            |            |            | |
| | 100     190         | |            |            :            :  OOO*******:****        |            |            | |
| |PUBLISH DETAIL DESIGN               |            :            :            :            |            |            | |
| | 130     170         | |            |            :            :    OOO***  :            |            |            | |
| |REVIEW FUNCTIONAL TEST PLAN         |            :            :            :            |            |            | |
| | 150     180         | |            |            :            :     OO*****:*           |            |            | |
| |REVIEW INTERFACE TEST PLAN          |            :            :            :            |            |            | |
| | 160     170         | |            |            :            :     ●●●●   :            |            |            | |
| |COMPLETE UNIT TEST| |               |            :            :            :            |            |            | |
| | 170     180         | |            |            :            :        ●●●:●           |            |            | |
| |COMPLETE FUNCTIONAL TEST            |            :            :            :            |            |            | |
| | 110     190         | |            |            :            :          O:O●**        |            |            | |
| |REVIEW SYSTEM TEST PLAN             |            :            :            :            |            |            | |
| | 180     190         | |            |            :            :            :●●●●        |            |            | |
| |COMPLETE INTERFACE TEST             |            :            :            :            |            |            | |
| | 190     200         | |            |            :            :            :   ●●●●●●●●●|            |            | |
| |COMPLETE SYSTEM TEST|               |            :            :            :            |            |            | |
| | 200     210         | |            |            :            :            :           ●|●●          |            | |
| |COMPLETE INSTALLATION               |            :            :            :            |            |            | |
| | 210     220         | |            |            :            :            :            | ●          |            | |
| |CONTINUE MAINTENANCE|               |            :            :            :            |            |            | |
-----------------------------------------------------------------------------------------------------------------------
| |                     |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                     | |    1972    |     10     :      20  19:73   30     :      40     |    1974    |    1975    | |
-----------------------------------------------------------------------------------------------------------------------
```

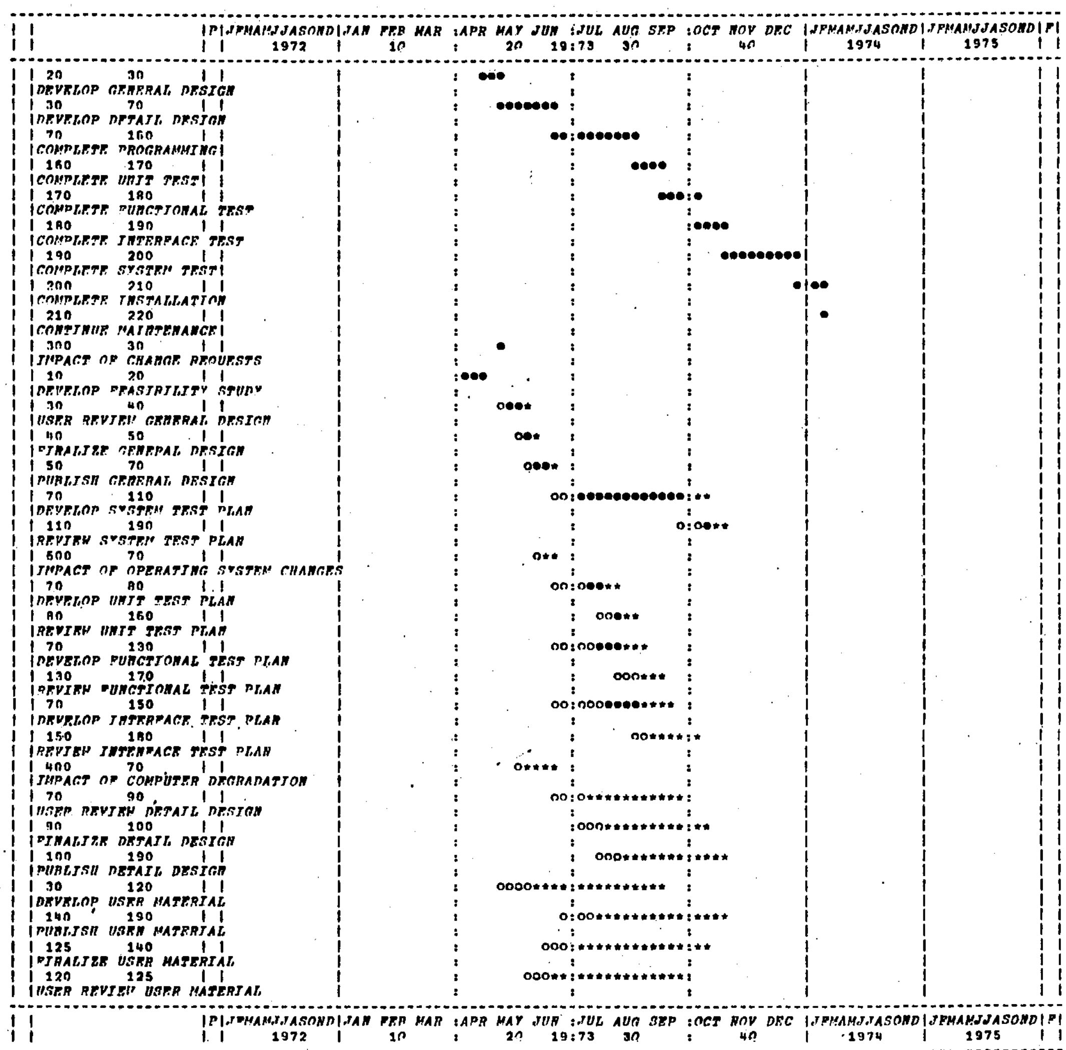
```
PROJECT:     M.I.S.  GENERAL MODEL                    PRINTED 04/05/73 AT 16.00.37

TIMENOW DATE USED:

SORTED BY LATE START

O-SPAN OF EARLY START TO EARLY FINISH
*-SPAN OF LATE START TO LATE FINISH
●-COMPLETED ACTIVITY EARLY START TO EARLY FINISH

---------------------------------------------------------------------------------------------------------------------------
| |                        |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                        | |    1972    |    10      :    20   19:73   30      :     40     |    1974    |    1975    | |
---------------------------------------------------------------------------------------------------------------------------
| | 10        20     | |   |            |            :●●●         :            :            |            |            | |
| |DEVELOP FEASIBILITY STUDY            |            :            :            :            |            |            | |
| | 20        30     | |   |            |            :  ●●●       :            :            |            |            | |
| |DEVELOP GENERAL DESIGN  |            |            :            :            :            |            |            | |
| | 30        70     | |   |            |            :   ●●●●●●●  :            :            |            |            | |
| |DEVELOP DETAIL DESIGN   |            |            :            :            :            |            |            | |
| | 300       30     | |   |            |            :   ●        :            :            |            |            | |
| |IMPACT OF CHANGE REQUESTS            |            :            :            :            |            |            | |
| | 30        40     | |   |            |            :   O●●*     :            :            |            |            | |
| |USER REVIEW GENERAL DESIGN           |            :            :            :            |            |            | |
| | 40        50     | |   |            |            :     O●*    :            :            |            |            | |
| |FINALIZE GENERAL DESIGN |            |            :            :            :            |            |            | |
| | 50        70     | |   |            |            :      O●●*  :            :            |            |            | |
| |PUBLISH GENERAL DESIGN  |            |            :            :            :            |            |            | |
| | 70        160    | |   |            |            :         ●●:●●●●●●●     :            |            |            | |
| |COMPLETE PROGRAMMING|   |            |            :            :            :            |            |            | |
| | 400       70     | |   |            |            :     O****  :            :            |            |            | |
| |IMPACT OF COMPUTER DEGRADATION       |            :            :            :            |            |            | |
| | 500       70     | |   |            |            :       O**  :            :            |            |            | |
| |IMPACT OF OPERATING SYSTEM CHANGES   |            :            :            :            |            |            | |
| | 70        110    | |   |            |            :         OO:●●●●●●●●●●●●:**          |            |            | |
| |DEVELOP SYSTEM TEST PLAN|            |            :            :            :            |            |            | |
| | 70        80     | |   |            |            :         OO:O●●**      :            |            |            | |
| |DEVELOP UNIT TEST PLAN  |            |            :            :            :            |            |            | |
| | 70        130    | |   |            |            :         OO:OO●●●***   :            |            |            | |
| |DEVELOP FUNCTIONAL TEST PLAN         |            :            :            :            |            |            | |
| | 70        150    | |   |            |            :         OO:OOO●●●●****:            |            |            | |
| |DEVELOP INTERFACE TEST PLAN          |            :            :            :            |            |            | |
| | 80        160    | |   |            |            :            :  OO●**     :            |            |            | |
| |REVIEW UNIT TEST PLAN   |            |            :            :            :            |            |            | |
| | 30        120    | |   |            |            :   OOOO****:**********  :            |            |            | |
| |DEVELOP USER MATERIAL   |            |            :            :            :            |            |            | |
| | 160       170    | |   |            |            :            :       ●●●● :            |            |            | |
| |COMPLETE UNIT TEST| |   |            |            :            :            :            |            |            | |
| | 130       170    | |   |            |            :            :     OOO*** :            |            |            | |
| |REVIEW FUNCTIONAL TEST PLAN          |            :            :            :            |            |            | |
| | 70        90     | |   |            |            :         OO:O***********:            |            |            | |
| |USER REVIEW DETAIL DESIGN            |            :            :            :            |            |            | |
| | 170       180    | |   |            |            :            :         ●●●:●          |            |            | |
| |COMPLETE FUNCTIONAL TEST|            |            :            :            :            |            |            | |
| | 120       125    | |   |            |            :      OOO**:************:            |            |            | |
| |USER REVIEW USER MATERIAL            |            :            :            :            |            |            | |
| | 150       180    | |   |            |            :            :      OO****:*           |            |            | |
| |REVIEW INTERFACE TEST PLAN           |            :            :            :            |            |            | |
| | 90        100    | |   |            |            :            :OOO*********:**          |            |            | |
| |FINALIZE DETAIL DESIGN  |            |            :            :            :            |            |            | |
| | 125       140    | |   |            |            :        OOO:************:**          |            |            | |
| |FINALIZE USER MATERIAL  |            |            :            :            :            |            |            | |
| | 180       190    | |   |            |            :            :            :●●●●        |            |            | |
| |COMPLETE INTERFACE TEST |            |            :            :            :            |            |            | |
| | 100       190    | |   |            |            :            :  OOO*******:****        |            |            | |
| |PUBLISH DETAIL DESIGN   |            |            :            :            :            |            |            | |
| | 110       190    | |   |            |            :            :           O:O●**        |            |            | |
| |REVIEW SYSTEM TEST PLAN |            |            :            :            :            |            |            | |
| | 140       190    | |   |            |            :           O:OO*********:****        |            |            | |
| |PUBLISH USER MATERIAL   |            |            :            :            :            |            |            | |
| | 190       200    | |   |            |            :            :            :  ●●●●●●●●●●|            |            | |
| |COMPLETE SYSTEM TEST|   |            |            :            :            :            |            |            | |
| | 200       210    | |   |            |            :            :            :           ●|●●          |            | |
| |COMPLETE INSTALLATION   |            |            :            :            :            |            |            | |
| | 210       220    | |   |            |            :            :            :            | ●          |            | |
| |CONTINUE MAINTENANCE|   |            |            :            :            :            |            |            | |
---------------------------------------------------------------------------------------------------------------------------
| |                        |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                        | |    1972    |    10      :    20   19:73   30      :     40     |    1974    |    1975    | |
---------------------------------------------------------------------------------------------------------------------------
```

PROJECT: M.I.S. GENERAL MODEL PRINTED 04/05/73 AT 10.16.56

TIMENOW DATE USED:

SORTED BY SLACK

O-SPAN OF EARLY START TO EARLY FINISH
*-SPAN OF LATE START TO LATE FINISH
□-COMPLETED ACTIVITY EARLY START TO EARLY FINISH



```
PROJECT:     M.I.S.  GENERAL MODEL                PRINTED 04/05/73 AT 16.34.47

TIMENOW DATE USED:

SORTED BY PRED-SUCC

O-SPAN OF EARLY START TO EARLY FINISH
*-SPAN OF LATE START TO LATE FINISH
Π-COMPLETED ACTIVITY EARLY START TO EARLY FINISH

---------------------------------------------------------------------------------------------------------------------
| |                  |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                  | |     1972   |     10     :     20   19:73   30     :     40     |   1974     |   1975     | |
---------------------------------------------------------------------------------------------------------------------
| | 10       20      | |            |            :●●●         :            :            |            |            | |
| |DEVELOP FEASIBILITY STUDY        |            :            :            :            |            |            | |
| | 20       30      | |            |            :  ●●●       :            :            |            |            | |
| |DEVELOP GENERAL DESIGN           |            :            :            :            |            |            | |
| | 30       40      | |            |            :    O●●*    :            :            |            |            | |
| |USER REVIEW GENERAL DESIGN       |            :            :            :            |            |            | |
| | 30       70      | |            |            :    ●●●●●●● :            :            |            |            | |
| |DEVELOP DETAIL DESIGN            |            :            :            :            |            |            | |
| | 30       120     | |            |            :    OOOO****:**********  :            |            |            | |
| |DEVELOP USER MATERIAL            |            :            :            :            |            |            | |
| | 40       50      | |            |            :      O●*   :            :            |            |            | |
| |FINALIZE GENERAL DESIGN          |            :            :            :            |            |            | |
| | 50       70      | |            |            :       O●●* :            :            |            |            | |
| |PUBLISH GENERAL DESIGN           |            :            :            :            |            |            | |
| | 70       80      | |            |            :          OO:O●●**       :            |            |            | |
| |DEVELOP UNIT TEST PLAN           |            :            :            :            |            |            | |
| | 70       90      | |            |            :          OO:O***********:            |            |            | |
| |USER REVIEW DETAIL DESIGN        |            :            :            :            |            |            | |
| | 70       110     | |            |            :          OO:●●●●●●●●●●●●:**          |            |            | |
| |DEVELOP SYSTEM TEST PLAN         |            :            :            :            |            |            | |
| | 70       130     | |            |            :          OO:OO●●●***    :            |            |            | |
| |DEVELOP FUNCTIONAL TEST PLAN     |            :            :            :            |            |            | |
| | 70       150     | |            |            :          OO:OOO●●●****  :            |            |            | |
| |DEVELOP INTERFACE TEST PLAN      |            :            :            :            |            |            | |
| | 70       160     | |            |            :          ●●:●●●●●●●     :            |            |            | |
| |COMPLETE PROGRAMMING|            |            :            :            :            |            |            | |
| | 80       160     | |            |            :            : OO●**      :            |            |            | |
| |REVIEW UNIT TEST PLAN            |            :            :            :            |            |            | |
| | 90       100     | |            |            :            :OOO*********:**          |            |            | |
| |FINALIZE DETAIL DESIGN           |            :            :            :            |            |            | |
| | 100      190     | |            |            :            :  OOO*******:****        |            |            | |
| |PUBLISH DETAIL DESIGN            |            :            :            :            |            |            | |
| | 110      190     | |            |            :            :           O:O●**        |            |            | |
| |REVIEW SYSTEM TEST PLAN          |            :            :            :            |            |            | |
| | 120      125     | |            |            :       OOO**:************:            |            |            | |
| |USER REVIEW USER MATERIAL        |            :            :            :            |            |            | |
| | 125      140     | |            |            :         OOO:************:**          |            |            | |
| |FINALIZE USER MATERIAL           |            :            :            :            |            |            | |
| | 130      170     | |            |            :            :    OOO***  :            |            |            | |
| |REVIEW FUNCTIONAL TEST PLAN      |            :            :            :            |            |            | |
| | 140      190     | |            |            :           O:OO**********:****        |            |            | |
| |PUBLISH USER MATERIAL            |            :            :            :            |            |            | |
| | 150      180     | |            |            :            :      OO****:*           |            |            | |
| |REVIEW INTERFACE TEST PLAN       |            :            :            :            |            |            | |
| | 160      170     | |            |            :            :      ●●●●  :            |            |            | |
| |COMPLETE UNIT TEST| |            |            :            :            :            |            |            | |
| | 170      180     | |            |            :            :         ●●●:●           |            |            | |
| |COMPLETE FUNCTIONAL TEST         |            :            :            :            |            |            | |
| | 180      190     | |            |            :            :            :●●●●        |            |            | |
| |COMPLETE INTERFACE TEST          |            :            :            :            |            |            | |
| | 190      200     | |            |            :            :            :   ●●●●●●●●●|            |            | |
| |COMPLETE SYSTEM TEST|            |            :            :            :            |            |            | |
| | 200      210     | |            |            :            :            :           ●|●●          |            | |
| |COMPLETE INSTALLATION            |            :            :            :            |            |            | |
| | 210      220     | |            |            :            :            :            |●           |            | |
| |CONTINUE MAINTENANCE|            |            :            :            :            |            |            | |
| | 300      30      | |            |            :    ●       :            :            |            |            | |
| |IMPACT OF CHANGE REQUESTS        |            :            :            :            |            |            | |
| | 400      70      | |            |            :      O**** :            :            |            |            | |
| |IMPACT OF COMPUTER DEGRADATION   |            :            :            :            |            |            | |
| | 500      70      | |            |            :        O** :            :            |            |            | |
| |IMPACT OF OPERATING SYSTEM CHANGES            :            :            :            |            |            | |
---------------------------------------------------------------------------------------------------------------------
| |                  |P|JFMAMJJASOND|JAN FEB MAR :APR MAY JUN :JUL AUG SEP :OCT NOV DEC |JFMAMJJASOND|JFMAMJJASOND|F|
| |                  | |     1972   |     10     :     20   19:73   30     :     40     |   1974     |   1975     | |
---------------------------------------------------------------------------------------------------------------------
```

VITA

Warren Daniel Nilsson

Candidate for the Degree of

Doctor of Philosophy

Thesis: TESTING A COMPLEX, REAL-TIME MANAGEMENT INFORMATION SYSTEM

Major Field: Engineering

Biographical:

Personal Data: Born in Green Bay, Wisconsin, March 6, 1942, the son of Mr. and Mrs. Warren O. Nilsson.

Education: Graduated from Robert E. Fitch Senior High School, Groton, Connecticut, in June, 1960; received Bachelor of Science in Industrial Engineering from University of Rhode Island, Kingston, Rhode Island, on June 7, 1964; received Master of Science in Engineering, with a major in Industrial Engineering, from New York University, New York, New York on June 9, 1965; completed requirements for the Doctor of Philosophy degree at Oklahoma State University, Stillwater, Oklahoma, in July, 1973.

Professional Experience: From 1965 to 1968 was an industrial engineer at International Business Machines (IBM) in Poughkeepsie, New York. Major responsibilities included work measurement, manpower planning, manufacturing area facilities planning, indirect area facilities planning, cost estimating, etc. Also provided industrial engineering consulting support (advanced industrial engineering) for the one-hundred industrial engineers in IBM Poughkeepsie. Most important contribution was the design and installation of IBM's longest and most complex conveyor system. From 1969 through 1971 was a systems analyst for IBM at Port Chester, New York and Mahwah, New Jersey. Major responsibilities included the coordination of various activities in the development test and installation of the world's largest civilian management information system.