1984

Logical design of a decision support system to forecast technology, prices and costs for the National Communications System

Williams, Kent A.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

LOGICAL DESIGN OF A DECISION SUPPORT SYSTEM TO FORECAST TECHNOLOGY, PRICES AND COSTS FOR THE NATIONAL COMMUNICATIONS SYSTEM

by

Kent A. Williams
Edwin C. Partridge III

September 1984

Thesis Advisor: Carl R. Jones

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE *(and Subtitle)*

Logical Design of a Decision Support System to Forecast Technology, Prices and Costs For the National Communications System

5. TYPE OF REPORT & PERIOD COVERED

Master's Thesis
September 1984

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

Kent A. Williams
Edwin C. Partridge III

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Naval Postgraduate School
Monterey, CA 93943

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

Naval Postgraduate School
Monterey, CA 93943

12. REPORT DATE

September 1984

13. NUMBER OF PAGES

111

14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*

15. SECURITY CLASS. *(of this report)*

Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Decision Support System; National Communications System; Forecasting

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This work describes the logical design of a proposed decision support system for use by the National Communications System in forecasting technology, prices and costs. It is general in nature and only includes those forecasting models which are suitable for computer implementation. Because it is a logical design it can be coded and applied in many different hardware and/or software configurations.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

Approved for public release; distribution unlimited.

Logical Design of a Decision Support System to Forecast Technology, Prices and Costs For the National Communications System

by

Kent A. Williams
Lieutenant, United States Navy
B.S., United States Naval Academy, 1977

and

Edwin C. Partridge III
Captain, United States Army
B.S., University of Southern Mississippi, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
September 1984

ABSTRACT

This work describes the logical design of a proposed decision support system for use by the National Communications System in forecasting technology, prices and costs. It is general in nature and only includes those forecasting models which are suitable for computer implementation. Because it is a logical design it can be coded and applied in many different hardware and/or software configurations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

The design and construction of software for a modern information system is a rigorous process which can be described in a life cycle which consists of the following steps:

1. Feasibility - Defining the basic approach and general scope of a software project, with particular emphasis on determining the practicality of such a project over its entire life span.
2. Requirements - Specifying the required functions, interfaces and actual performance of the software system, including operational constraints.
3. Design - Defining data flow, algorithms, data representations and control structures. Identifying and specifying modules. Usually entails at least two iterations of refinement.
4. Coding - Translating the design into a programming language. Includes testing of individual components.
5. Integration - Individual system components are integrated into the final system configuration.
6. Implementation - Installation of the software product with the host hardware system, to include testing.
7. Maintenance - All subsequent alterations, modifications and improvements made to the complete system.

This work is an attempt to define a logical software design, based on general system requirements, which can be "fine-tuned" by rigorous specification and ultimately used as the foundation for the physical implementation of a decision support system (DSS). As such, it does not strictly follow the prescribed conventional software development process. It is, rather, a hybrid approach at addressing

several of the traditional phases of the life cycle: feasibility, requirements and (logical) design. It is purposely of such a general nature in order to facilitate specific system requirements and ultimate performance of the entire software development life cycle.

One of the underlying design principles upon which this effort is based is that of software generality. An automated system to forecast telecommunications technology, prices and costs should be designed in such a manner as to allow for maximum utility within the proposed scope of application. This particular DSS logical design enables the NCS manager to model a wide variety of technology, price and cost situations without the associated overhead imposed by multiple application-specific systems.

The Manager of the National Communications System (NCS) has been tasked by the National Security Telecommunications Policy of 3 August 1983 with implementing this policy under the direction and with the consultation of the Policy Steering Group. As part of this task, the Manager must:

1. Ensure the development, in conjunction with the NCS operating agencies, of plans to fulfill the principles and objectives stated in this directive, including an overall telecommunications architecture and timetable.
2. Develop, for review by the Steering Group, overall budget profiles regarding approved initiatives and related activities.
3. Prepare annually, or as otherwise directed, a written report to the Steering Group on the progress of approved initiatives, including an assessment of the resources that will be required to attain the objectives of this directive [Ref. 1].

For a manager to make effective decisions and to prepare effective and timely plans, a certain amount and quality of

information has to be available. Martino [Ref. 2] states, "One of the best-kept secrets of the planning profession is that planning has nothing to do with actions to be taken in the future. Instead, planning deals with actions to be taken in the present."

The information required for planning into the future partly consists of estimates as to what the future holds. The NCS must have estimates of what technologies will be available at a later time and what the cost and price of that technology will be. A decision support system system which utilizes forecasting techniques and models can be used to derive accurate estimates and aid NCS managers in making decisions based upon these estimates. Forecasts of technology are useful because a technological change can:

1. Provide new methods of achieving objectives.
2. Render certain means of achieving objectives obsolete.
3. Render certain objectives obsolete.

The necessity to track technology growth is particularly important once it has been identified as a needed technology. Isenson [Ref. 3] found through his investigations of Project Hindsight that a real need results in accelerated technological growth. The greater the rate of the growth of a technology, the more it can influence previously made plans.

This paper will develop a decision support system to forecast technology, prices, and costs for use by the National Communications System. The next chapter is an overview of the NCS. Chapter III introduces the concept of forecasting and forecasting models, with a discussion on how they may effectively be utilized by a government agency such as the NCS. The actual logical design for the decision support system will then be developed and the methods utilized in its design are discussed. The conclusions will

describe why the particular forecasting models implemented in the DSS were selected and also discuss possible strategies for implementation of the DSS.

II. THE NATIONAL COMMUNICATIONS SYSTEM

A. OVERVIEW

The National Communications System was formed on 21 August 1963 as a result of a recommendation by the National Security Council that the Executive Office move to identify and coordinate the communications needs of the Federal Government. Originally envisioned as a means to integrate the many systems found throughout the government, the general mission of the NCS continues to be to ensure the surviveability of communications during and subsequent to any national emergency. In order to accomplish this mission the NCS is organized not as a homogenous, separate entity; rather, it is an arrangement of heterogeneous telecommunications systems which are provided by their sponsor Federal agencies. In its early years of existence the NCS was comprised mostly of General Services Administration (GSA) and Department of Defense (DOD) assets. Today, however, virtually every major Federal agency is a participating member of the NCS.

The physical components of Federal telecommunications systems and networks included under the NCS may be described as the following:

1. Automatic telephone route control switching facilities and associated first level user switching facilities.
2. Telephone and digital data switching facilities and primary common user communications centers.
3. Special purpose local delivery message switching and exchange facilities.

4. Fixed route government owned or leased transmission facilities under exclusive control of a government agency.
5. Government owned or leased radio systems.
6. Technical control facilities which are under exclusive control of a government agency.
7. Other services provided by a common or specialized carrier on a continuing basis, via commercial facilities not designated for exclusive government use. These services, like exclusive use services, will still be assigned an appropriate restoration priority in the event of national emergency or other disruption of the service. [Ref. 4]

Most NCS operating component systems are long-haul, trunk, point-to-point systems. They are planned, operated and funded by their sponsor agencies to fulfill a specific peacetime need. The current NCS management doctrine is to provide joint central planning, standardization and programming. The long range goal is to ensure progressive, systematic improvements existing systems in order to allow efficient and effective transition from peacetime to emergency conditions.

B. NCS POLICY

As the number of NCS operating components has grown over the years, so also have the organizational responsibilities and system complexities. In order to clarify and define the NCS goals and objectives, the National Security Council (NSC) established in 1979 the National Security Telecommunications Policy. This policy directed the NCS to ensure telecommunications assets provide for:

1. Emergency communications between the National Command Authority and appropriate forces to support retaliatory action in the event of enemy nuclear attack.

2. Military operational communications for both general and nuclear conflicts.
3. Communications in support of military mobilization.
4. Communications to ensure government continuity in the event of nuclear or natural disaster, and recovery from the same. [Ref. 5]

In August 1983 the national telecommunications policy was further defined. The new policy addresses the vulnerability of existing NCS systems to nuclear attack and directs enhancements and improvements (i.e., switches and control centers) be physically located away from likely nuclear target areas and, whenever feasible, existing system components be hardened. [Ref. 1]

Actual policy guidance for the NCS in the areas of telecommunications planning and development is fragmented and originates from multiple sources at the Executive Office level. For example, the Office of Science and Technology Policy (CSTP) is tasked with the responsibility for the collection, recording and evaluation of existing telecommunications facilities and the development of profile information detailing the residual capabilities of these systems and networks under various extraordinary conditions, including nuclear and other national emergencies [Ref. 6]. Such information is used by the NCS in the conduct of restoration and allocation activities, resources evaluation, damage assessment, requirements evaluation, and priority determination. The OSTP facilities status evaluation provides the NCS raw data pertaining to system gross operational capabilities in terms of (1) link/trunk capability, (2) call demand/acceptance capability of voice switching systems, (3) message processing capability of record traffic switching systems, (4) user/subscriber access capability of voice and record switching systems, and (5) residual major system access concentraters. This data can and should be

utilized by the NCS to ensure Federal telecommunications systems, derived from common carrier networks, are interconnected and capable of interoperation to the maximum extent possible.

C. PROCUREMENT OF SERVICES

More than 95% of the communication services utilized by the Federal government and its agencies within the continental United States are provided by common user systems leased from and operated by the major common and specialized commercial carriers (the vast majority are leased from AT&T Long Lines and associated Bell operating or interconnect companies) [Ref. 7]. "Common" user systems means that the physical facilities from which the government services are derived are usually also common to public message services with provisions made to segregate the two services. Close and continual coordination between the NCS and the private sector telecommunications industry is facilitated by the Federal Communications Commission (FCC) and the National Industry Advisory Committee (NIAC). During wartime or other national emergency the authority to requisition and contract for supplies and equipment, and to restore, expand, repair and construct telecommunications systems is delegated to the FCC. However, the NCS retains overall responsibility for integrating all government communications. In order to perform its emergency functions, the FCC relies heavily upon the NCS Telecommunications Emergency Management System (NTEMS).

Peacetime procurement responsibility is centralized also, but divided between GSA for civil agencies and the Defense Communications Agency (DCA) for DOD systems. Certain civil agencies and components have been authorized independent authority to procure directly from common and

specialized carriers where it has been determined to be more convenient for the government. Such exceptions to the rule are rare, however, primarily due to the paramount necessity to ensure mutual support and interoperability among the various systems.

III. USE OF FORECASTING MODELS IN GOVERNMENT

A. CONCEPTS

The general concept of cost and price projection is an integral issue associated with the acquisition of major systems, commercial products and industrial services by agencies of the federal government. This concept normally is addressed during the rigorous process known as economic analysis, the outcome of which is a major factor either in the selection of a choice between two or more alternatives, or in assessing the economic consequences of a choice already made between alternatives. Unfortunately, the literature pertaining to economic analysis includes rather generalized and imprecise guidance for the conduct of cost/price estimation, key elements of the process. In practice, costing and pricing within the federal government varies widely from agency to agency, and even within agencies there may be variety, dependent upon the type of product or service being acquired [Ref. 8]. The use of technology, price and cost forecasting models is one method available to complement and enhance an agency's established emplcyment of economic analysis and estimation in the acquisition life cycle.

Regardless of the scope of the project or program involved, the acquisition process can be viewed as a logical progression of iterative reviews, determinations and evaluations to reconcile periodic adjustments to program objectives and requirements or resource availability. This process overlaps the traditional functions of planning, budgeting, contracting and contract administration, each of which can be examined as an area of specialization.

Historically, federal departments and agencies have utilized separate estimation and analysis units within each stage of the acquisition process. In addition, each unit has tended to utilize a unique costing and pricing technique for each of the functional specialties. Furthermore, as each estimate and analysis is forwarded through the organizational hierarchy, policy reviews and revisions take place. Although some agencies utilize a centralized cost/price estimation and analysis activity, they are the exception to the rule. The normal process entails redundancy of effort and, all to often, results in poor cost and price information. Certainly it is given that the adequacy of data is a major factor in the quality of cost and price estimates and analyses, but the importance of the overall methodology utilized must not be discounted. This is the case particularly for estimates and analyses involving new technology and major systems.

The traditional acquisition costing/pricing process can be significantly enhanced by use of forecasting techniques and methods. Forecasting is a process whose objective is to predict future events or conditions under an assumed set of circumstances. The most common applications of forecasting involve the use of estimating models to predict quantitative values of certain variables outside the sample of data actually observed. In the case of cost and price forecasts, these values would most likely assume a probability distribution rather than a point forecast. Technology forecasting looks more toward the time period by which certain parameters of a given technology will be achieved. An example of this would be to forecast the year in which 90% of telecommunications common carriers will be using digital voice circuits rather than analog circuits.

The forecasting process, like the product or service for which the forecast is made, can be described in terms of a

life cycle. The forecaster begins the process by identifying facts and other data about past trends and previous forecasts relating to the problem under consideration. Particular attention is given to determining the cause of variances between previous forecasts and actual system behavior. The forecaster must next determine and organize future parameters of the decision problem. A suitable model which describes the problem space is then constructed, along with a method and measure of accuracy and reliability. During the course of the project, periodic samples are taken to compare the forecast with actual behavior, documenting variances as they occur. Finally, the forecast is revised as necessary. [Ref. 2]

A forecast is only as accurate as its model, and a model is only as accurate as its data sources. Moreover, the model used represents a compromise between reality and manageability. It must identify essential factors while disregarding non-critical ones. A good model specifies interrelationships among parts of the system such that it is reasonably detailed and explicit to ensure the model adequately describes the real-world system. However, it must also specify them in such a way that it is understandable so that proper analysis and conclusions regarding the real-world system can be made.

B. PRICE/COST FORECASTING MODELS

This work is not an attempt to survey the entire field of available forecasting models. The focus will be on the most common type of parametric model, the algebraic model, which is particularly well suited to the NCS application because of the ease with which it may be expanded and modified. The algebraic model typically consists of several equations, each with a separate meaning and role. The model

determines values of certain endogenous variables, the jointly dependent variables which are simultaneously solved by the relations of the model. Independent exogenous variables are determined outside the system but influence it by affecting the values of the endogenous variables. They affect the system but are not in turn affected by it. An econometric model is an algebraic model that typically includes one or more random variables (disturbance terms) and represents a system by a set of stochastic relations among the variables of the system. Such a model generally specifies the probability distribution of each endogenous variable, given the values taken by all exogenous variables and given the values of all parameters of the model. [Ref. 9]

A cost model is used to predict the anticipated costs likely to be incurred in a project. Like other models, when a cost model equation or system of equations is derived from statistical analysis of a sample of past projects, an associated factor is a degree of imprecision or uncertainty. The validity of the model is a function of how widely the data are scattered around the prediction line or curve. Cost models must generally be individually structured to best meet the purpose for which they are intended (Table 1). If the forecast is meant to aid in the choice between alternatives, the differential life cycle cost model would be used. Such a model would compare the differential costs associated with the alternative systems. Detailed comparison between the alternatives would be provided by summing the differential costs identified with the applicable cost elements chosen. Conversely, a cost model based upon total life cycle cost would concentrate on applicable cost elements over the projected life expectancy of a particular equipment or system. The model builder would identify the cost categories and associated cost elements to be utilized

TABLE 1

Life Cycle Cost Models

Total Life Cycle Cost Model

Associates all applicable cost elements over the lifetime of a system.

Differential Life Cycle Cost Model

Compares differential costs between two similar cost elements of two different systems.

as model parameters. Table 2 is an explanation of the conventional major cost categories.

C. TECHNOLOGY FORECASTING MODELS

Technology forecasting models are similar to price/cost models in that the primary determinant of the quality of the forecast is in the variables which are brought into the model and how they are weighted in relation to one another. Once this has been accomplished, different techniques can be utilized to fit a curve to the historic data in order to project the value of the technology parameter at a future time. The model is highly dependent upon the core assumptions made about the environment which is being forecast. In particular is the assumption regarding the existence of upper limits (or lower limits in the case of time reductions) on the capabilities of the technology being modeled. An example of an upper limit assumption would be the speed of light for spacecraft velocities. Other than extrapolating trends into the future by curve fitting, technology forecasting can sample the amount of literature circulating in an area of technology. By monitoring the growth (or lack

TABLE 2

Major Cost Categories

Research and Development Costs

All costs associated with the research, development, test and evaluation of the equipment/system. Normally these costs are incurred during concept initiation, validation and full scale development.

Nonrecurring Investment Costs

All costs incurred one time beyond the program development phase and during the program production phase. These costs can occur if there is a change in design, contractor or manufacturing process.

Recurring Investment Costs

All production costs that recur with each unit produced.

Operating and Maintenance Costs

All costs associated with personnel, material, facilities and other costs required to operate, maintain and support an equipment/system during its useful lifetime.

of growth) in an area, it can be estimated that the increased communications among researchers will result in an exponential growth of knowledge, likely to result in a breakthrough or an advancement of the technology being studied. This area of forecasting can be directly influenced by infusion of government resources into research [Ref. 3]. If a certain level of a parameter of a technology is desired by a certain date, the amount of research and development necessary now can be estimated. The recent Presidential initiative regarding the so-called "Star Wars" technology is an example of this technique.

D. SUMMARY

Although forecasting models are well suited for adaptation to automated systems, they are not without their potential problems. Depending upon the real-world project and model application, forecasters may find a limited number of relevant statistical procedures available. Once constructed, models may quickly become obsolete by the rapid growth of the technology being forecast. Hence, effort must be directed toward a method for adapting the model for technological advance even during periods of rapid growth. The forecaster may be confronted with the mathematical problem of solving k equations in n unknowns ($k < n$). A host of problems involving accuracy of the model may be caused by omission of a relevant exogenous variable, disregarding a qualitative change in one of the variables, inclusion of an irrelevant variable, incorrect definition of a variable, and in the case of econometric models, incorrect specification of the manner in which the stochastic disturbance term enters the equation.

The effects of divestiture and deregulation of the telecommunications industry are major contributing reasons for the National Communications System to consider use of forecasting techniques. As the NCS continues to grow in size, scope and complexity of participating systems (for example, the conversion from analog to digital voice circuits), even more powerful tools will be required to exert effective managerial control over further development. Accordingly, the objective of forecasting technology, cost and price is not to provide a managerial decision, but to derive further inputs to the managerial decision-making process. Numerous potential applications exist within the traditional Planning, Programming and Budgeting System (PPBS) and acquisition cycles. For example, it can be used in lieu of

conventional cost-estimating during the cost/benefit analysis (concept development of the acquisition cycle). It can be used to evaluate alternatives during strategic planning. It can be utilized during periodic revisions of the Five-Year Defense Plan (FYDP), or similar intermediate-term plan for GSA-procured systems. As a general rule, an automated forecasting system would be ideal for use whenever traditional economic/technological analysis is too elaborate, too time-consuming and/or too expensive for the scope of the particular problem or project at hand.

IV. PRELIMINARY DATA DICTIONARY DESIGN

In order to develop a database for a decision support system it is necessary to look at the overall requirements for designing such a system with special emphasis on the data which will be utilized. The DSS architecture presented by Dolk [Ref. 10] consists of four major components: (1) dialog, (2) model base, (3) knowledge base, and (4) database. The dialog is the primary driver of the system. It is the interface with the user; therefore, the dialog is dependent upon what outputs the decision-maker wants from the DSS and what inputs can be to provided to get that output. The model base will provide the basic algorithms of the system models as well as the value abstractions of the coefficients for the variables to be utilized by the model algorithms. The knowledge base contains a set of heuristics which determine what type model or combination of models will be processed for a given circumstance provided by the user. The database will contain the structure and values of all data in the DSS which is subject to modification and/or addition by the user without modifying the program itself. The data utilized by the dialog, model base, and knowledge base determine what will be in the database, and will therefore be examined briefly in turn.

A. DIALOG

The "rule of thumb" in DSS design (or in any systems analysis for that matter) is to first determine what will be the outputs and inputs to the system. Because all interface with the user is through the dialog, this is paramount to determining what the dialog is to be. The prime question to

be asked is, "What does the user need in a forecast of technology?" The answer to that question for the purposes of this DSS is that the decision-maker wants the DSS to provide a presentation of trends for a given parameter in an area of a technology, given a set of assumptions and optionally given a set of parameters from other areas which may impact upon the technology, including the degree to which they do so.

The following are outputs required from the dialog:

1. A list of the assumptions used in generating the forecast.
2. The type of model(s) being utilized.
3. A graphical presentation of historical data versus time extrapolated by one of the models to get an indication of a trend, whether increasing, decreasing or steady and which includes the scale used, whether linear or logarithmic.
4. The source of the data on the graph and its type, whether subjective, objective or estimate.
5. A comparison of this forecast with previous forecasts.
6. Which parameters of the model are exogenous or endogenous and of these, which can be influenced by the decision-maker.

The following inputs to the dialog are required:

1. An ability to create data with these characteristics: identifiers for name, type of factor it is, source of the data, date of the data entry and date of the data observation.
2. An ability to alter assumptions and parameters in order to observe any changes in the output. This can include a means for indicating the stochastic nature of some of the variables. For example, in a price/cost model the expected future interest rates of

treasury bonds may be estimated as a triangular distribution of the interest rate around a most likely value for the interest rate, bounded by an estimated high value and low value which can then be iterated through the model as a Monte Carlo simulation.

3. An ability to create different model equations for input into the model algorithm.
4. An ability to override the knowledge base and select a specific model to run.

B. MODEL BASE

This DSS utilizes two primary models. The first of these is a curve fitting model which regresses a straight line on the plots of five different functions of the factor or aggregate model score to forecast. These functions are a Pearl growth function, a Gompertz growth function, a function in which the natural logarithm of the dependent variable is taken, and a function in which the natural logarithm of the dependent variable and the natural logarithm of the independent variable are calculated. The regression which has the highest correlation factor is selected for use in extrapolating into the future. This method of forecasting has been selected due to its simplicity and intuitive understanding of the process by a manager. The second model in the DSS is a simple cross impact analysis model developed by Julius Kane [Ref. 2].

1. Scoring Model

A scoring model will take different factors named by the decision-maker and combine them to determine an aggregate score (hence the name scoring model). This is accomplished through queries directed at the user to determine

the relationships among the factors. Any factors which are essentially the same are eliminated so that only one factor will represent that area in the model. Next the factors are grouped to determine whether they are additive or multiplicative, either of the entire model, or of groups, and so on. Care must be taken in choice of multiplicative factors, for if the value of the factor is zero, then all of the factors which it multiplies are then zero. Weights are now assigned by the user to the different groups or individual factors. Desirable and undesirable factors and groups are separated with the desired factors being in the numerator and unfavorable factors being placed in the denominator. This is the basic model equation and can be stored as such.

The user must identify whether the data is subjective or objective. If subjective, the user will utilize a standard scale of zero to nine in selecting the value for the factor, while objective data will be examined and the mean and standard deviation for each factor's data being calculated. The mean of the data can then be assigned a value of the user's choice, and the other values determined as fractions of the standard deviation to range from a low to a high value also of the decision-maker's choice.

This type of model is useful in comparing different technologies which perform similar missions. An example drawn from telecommunications technology is a comparison of satellite communications versus landlines versus microwave links. For determining the relative vulnerability of each to disaster or nuclear attack, a subjective factor can be utilized, while cost of maintenance or installation will be an objective factor.

2. Pearl and Gompertz Curves

These two curves are discussed jointly because they are essentially the same functions, differing mainly in the

underlying assumptions. Both curves are "S" shaped and are extrapolated by first straightening out the curve as plotted by the factor data. This is accomplished by taking the logarithm of the curve, once for the Pearl curve, twice for the Gompertz curve. The data thus transformed has a straight line regressed on it to obtain the values for the curve functions. The equation[1] for the Pearl curve (Equation 4.1) and its algebraic transformation (Equation 4.2) utilize ln a as the constant and b as the slope, b taken to be positive. L is the assumed limitation of the technology and t is time while y is the value of the factor

$$y = L/(1 + a * (e ** (-b * t))) \qquad (4.1)$$

$$Y = (L - y)/y = \ln a - b * t \qquad (4.2)$$

under consideration. The Gompertz curve equation (Equation 4.3) and its algebraic transformation (Equation 4.4) utilize ln b as the constant and k as the slope. The other two

$$y = L * (e ** (-b * (e ** (-k * t)))) \qquad (4.3)$$

$$Y = \ln (\ln (L/y)) = \ln b - (k * t) \qquad (4.4)$$

variables are the same as before.

The different assumptions underlying the choice of these two curves is in the dynamics of the technology being forecast. If the previous progress in implementation or development of a technology will influence the rate of the progress of the technology, then a Pearl curve should be

---

[1]The following translations describe notations which may be unfamiliar to the reader:
'*' = multiplication operator
'**' = exponentiation operator
'ln' = natural logarithm function

used. However if the determining factor is how much remains to be accomplished before the assumed limit to growth of the technology is reached, then the Gompertz curve should be utilized.

An example of the use of the Pearl curve is a forecast of the number of households which have access to a broadband communications media. The factor in this instance is the number of homes with cable television installed. The maximum limit ('L') is that 100% of households which have televisions have cable installations. A Pearl curve is appropriate because the technology is driven by the degree of acceptance with which it is received by the public.

A forecast of the percentage of common carrier local distribution systems which will have optical fiber as the transmission media can be modeled by a Gompertz curve. The limit in this example is for 100% of existing local distribution systems to have been replaced by optical fiber systems. Since this substitution is influenced more by the number of systems remaining to be upgraded rather than by the number of systems which have already been implemented, a Gompertz curve is the correct growth curve for the forecast.

3. Cross Impact Analysis

This is a simple model which takes a factor in an area of a technology and determines the next value it will have as a result of the impact of other factors, both exogenous and endogenous. The model is simple in that only the impact of a single variable upon another variable is determined at a time, not all variables at once. The inputs by the decision-maker are purely subjective evaluations of what the impacts of certain chosen variables are upon the factor being evaluated, plus the original value of this factor (subjectively scaled from zero to one in increments of 0.001). The use of such a model is for a decision-maker to

get an idea of the results of varying the impacts of other factors upon a factor. Therefore, it is necessary for the impacting factors to be defined as either endogenous or exogenous variables so that the decision-maker will know which variables are able to be influenced. An example of an application of this model is found in Chapter VI.

C. KNOWLEDGE BASE

The knowledge base of this particular DSS does not have anything stored in the database. The rules for determining which models to run are within the algorithms of the program itself. The only impact upon the data dictionary is in the identification of objects passed by the dialog to the knowledge base. This would consist of determining whether a request for a model run is for more than one specific area of a technology, which would activate a scoring model to arrive at a conglomerate representation of the overall technology, or in determining whether a Pearl or Gompertz curve is to be utilized in extrapolation of the data. The cross impact model will be invoked when the user specifically directs that it be run.

D. DEVELOPMENT OF THE DATA DICTIONARY

The technique to be utilized in the development of this DSS is drawn from the works of Yourdon [Ref. 11] and DeMarco [Ref. 12]. Their methods of structured analysis and design result in a logical flow toward a complete software design without the large amounts of paper normally associated with a software design project. The less documentation which has to be changed in later design revisions of the DSS, the greater the possibility that the documentation will be updated to reflect changes made to the actual software. The essential elements of the DeMarco and Yourdon methods are

the development of a data flow, a data dictionary, and the process descriptions. In order to develop the data flow, the composition of the data inputs and outputs to the system have to be described in the data dictionary. The data flow will only show the flow of the data objects through the system, while the process descriptions provide information about the content and processing of the data.

The data objects are described in the data dictionary primarily through the use of the three types of relations presented by Bohm and Jacopini [Ref. 13]. These are sequence, selection and iteration. Sequence is a concatenation of two or more data objects and/or data elements together. Selection is a choice between two or more data items. Iteration is the repetition of a data object, or group of data objects, zero or more times. In addition to these relations, an optional relation is added so that it is possible to indicate if a data object may or may not be part of a larger data object. For this data dictionary a data element is considered to be data which is not further broken down into other data elements. The level to which a data element is broken down is left to the user and the data dictionary designer. A data object may be broken down into component data objects and/or data elements. Table 3 explains the notation utilized in this data dictionary.

The data dictionary for the DSS is developed by analyzing the descriptions of the dialog, model base and knowledge base in the previous sections. This will result in an initial look at how data objects may flow through the system. Because this is the initial version of the data dictionary it is inevitable that the data objects will change, be added, or be dropped if it appears during further design of the system that they will not be utilized by the system. Usually a data flow technique is utilized in analyzing an existing system in order to automate it. This

TABLE 3

Data Dictionary Notation

Notation	X Consists Of
x = a + b	data objects a and b
x = [a|b]	either a or b
x = {a}	zero or more occurances of a
x = (a)	optional data element a
x = y{a}	y or more occurances of a
x = {a}z	z or fewer occurances of a
x = y{a}z	between y and z occurances of a

design is different in that an attempt is being made to design a system which does not yet exist. Therefore the data dictionary depicted in Table 4 is admittedly of a preliminary nature.

With the use of this data dictionary an initial data flow can be constructed. The data flow will be expanded to different levels until the transformation of the input data to the output data is fully described. Upon the completion of the expansion of the data flow the process descriptions will be written. These descriptions will be compared with the data dictionary in order to determine if any of the data is not utilized or if there is data which must be added to the data dictionary. The data is then normalized and a data structure diagram is constructed. With the addition of the format in which the data will actually be stored, the data dictionary will be complete and serve as a reference document during the detailed design of the decision support system.

TABLE 4

Preliminary Data Dictionary

```
ALGORITHM_NAME      = *Name of a modeling algorithm
                      used as part of key to
                      identify a data object which
                      contains the data which will
                      be used by a model*

CHARACTERISTIC      = ["Exogenous"|"Endogenous"]
                      *Identifies whether data is
                      controllable*

DATE                = "MM/DD/YY"

DATE_OF_ENTRY       = DATE
                      *Date data entered into
                      database*

DATE_OBSERVATION    = DATE
                      *Date data for ELEMENT_ENTRY
                      observed or estimated*

DISTRIBUTION        = ["Norm"|"Uni"|"Tri"]
                      *How stochastic variable
                      is distributed*

ELEMENT             = *Sub-area within a
                      technology*

ELEMENT_ANALYSIS    = ["Historical"|"Estimate"]

ELEMENT_ENTRY       = DATE_OF_OBSERVATION
                      ELEMENT_SOURCE
                      DATE_OF_ENTRY
                      ELEMENT_VALUE
                      ELEMENT_ANALYSIS

ELEMENT_SOURCE      = *Source of data for
                      ELEMENT_ENTRY*

ELEMENT_VALUE       = MOST_LIKELY_VALUE
                      {HIGH_VALUE
                      LOW_VALUE
                      DISTRIBUTION}

FACTOR              = FACTOR_IDENTIFIER
                      FACTOR_TYPE
                      {UNITS}
                      CHARACTERISTIC
                      {ELEMENT_ENTRY}
```

Table 4

Preliminary Data Dictionary (cont'd)

FACTOR_IDENTIFIER	=	TECHNOLOGY ELEMENT
FACTOR_OPERATOR	=	GROUP_OPERATOR
FACTOR_TYPE	=	["Subj"|"Obj"]
FACTOR_WEIGHT	=	*Any positive integer - a subjective evaluation of the factor in the sub-group*
GROUP_IDENTIFIER	=	*A unique name within the data object to identify a grouping*
GROUP_LEVEL	=	*An integer greater than 0 used to indicate which other groups this group acts on The lower number groups act on all groups of higher number*
GROUP_OPERATOR	=	["Mult"|"Add"]
GROUP_WEIGHT	=	*Any real number - a subjective valuation of the group in the model*
GROUPS	=	GROUP_IDENTIFIER GROUP_OPERATOR GROUP_WEIGHT GROUP_LEVEL {SUB_GROUP SUB_GROUP_WEIGHT SUB_GROUP_OPERATOR}
HIGH_VALUE	=	*Any real number greater than or equal to the ELEMENT_VALUE's MOST_LIKELY_VALUE*
IMPACT_FACTOR	=	*A subjective value - a single digit 0-9*
IMPACTING_FACTOR	=	FACTOR_IDENTIFIER
LIMITING_FACTOR	=	FACTOR_IDENTIFIER FACTOR_TYPE {UNITS} CHARACTERISTIC 1{ELEMENT_ENTRY}1

Table 4

Preliminary Data Dictionary (cont'd)

```
LOW_VALUE             =  *Any real number less than or
                         equal to the ELEMENT_VALUE's
                         MOST_LIKELY_VALUE*

MODEL                 =  ALGORITHM_NAME
                         MODEL_IDENTIFIER
                         {GROUPS}
                         {LIMITING_FACTOR}
                         {IMPACTING_FACTOR
                          IMPACT_FACTOR}

MODEL_IDENTIFIER      =  *Name given by user used
                         along with ALGORITHM_NAME
                         to identify the data object
                         containing the data to
                         be modeled*

MOST_LIKELY_VALUE     =  *Any real number*

SCALE_FOR_DATA        =  ["Linear"|"Log"]

SUB_GROUP             =  SUB_GROUP_IDENTIFIER
                         {SUB_GROUP
                         SUB_GROUP_WEIGHT
                         SUB_GROUP_OPERATOR}
                         {FACTOR
                         FACTOR_OPERATOR
                         FACTOR_WEIGHT}
                         *A recursive definition
                         Sub-groups may contain
                         other subgroups*

SUB_GROUP_IDENTIFIER  =  *A unique name for a sub-
                         group within the group*

SUB_GROUP_OPERATOR    =  GROUP_OPERATOR

SUB_GROUP_WEIGHT      =  *Any positive integer - a
                         subjective evaluation of the
                         SUB_GROUP in the GROUP or
                         SUB_GROUP*

TECHNOLOGY            =  *Name of a technological area
                         at user's discretion*

UNITS                 =  *Units of measure for
                         ELEMENT_ENTRY's*
```

V. PROCESS SPECIFICATIONS

The process specifications are descriptions of each of the nodes in the system where data flows are transformed from one form of composition into another composition. A characteristic of these specifications is that each should describe an underlying policy of the system, specifying what is to be accomplished rather than how to accomplish it. To do this they are written in a form known as Structured English. Structured English is a form of English in which the majority of nouns used will come from the data dictionary. A reserved list of words is utilized to denote the actions within the process. Examples of words from this list are those words which use the three techniques of program construction. For sequence structures statements within a program should follow one another. To show decision the usual constructs are 'If...then...else', 'If...then', or 'If...then...otherwise'. For multiple decisions some variation of a 'Case' structure is employed (i.e., Case of This, Do This, Do That, Do The Other Thing). Iterations are expressed as 'Repeat...Until some condition is met','While a condition is present do...', or 'For a certain number of times do...'. A thorough treatment of the topic is provided in [Ref. 12].

The process specifications written here are referred to as process mini-specifications or 'mini-specs', due to the fact that each specification is unique to itself and describes a smaller system contained within the whole system, each having its specified inputs and outputs. To the remainder of the system the process will appear to be a 'black box' with inputs going in and outputs coming out, somehow transformed by the process. In this chapter the

inputs and outputs for each process are listed. The underlying policy of each process is provided to indicate what the process is to do. Due to the length of the mini-specs, they have been placed in a separate appendix. Prior to each mini-specification the inputs and outputs along with their respective sources and destinations are provided. Following the mini-specifications are the McCabe complexity numbers of the processes, and a description of the basis paths of the processes.

A. THE MCCABE COMPLEXITY METRIC IN SOFTWARE DESIGN

The McCabe Cyclomatic Complexity Measure was first developed for testing of already coded modules. McCabe's paper [Ref. 14] presents the idea of applying a complexity measure in the design phase of software design. Previously this metric had only been applied to completed code. The reasoning behind application of this metric in the design phase is that many more errors occur in the design phase than in the coding phase. This fact is demonstrated by the

TABLE 5
Relative Frequency of Design and Coding Errors

Modification	Source Statements (No.)	Design Errors (%)	Coding Errors (%)
A	1253	73.6	26.4
B	9880	73.7	26.3
C	779	35.6	64.4
D	9631	51.6	48.4
E	4575	58.8	41.2

data in Table 5 which is from a software reliability study conducted at TRW of the percent of errors introduced in a series of modifications to a large software project (100,000 lines of code) [Ref. 15]. The extension of the McCabe

complexity metric to the design testing of processes will help to identify logic path errors and will provide the number of basis paths through a process. A basis path is one of the set of possible independent paths from the entry point of the process to the exit point from the process. The set does not include variations from the independent paths due to iterations of statements along the path. Knowledge of these paths helps to determine the makeup of the test data to be utilized later in testing the coded designs.

The primary purpose of the McCabe Cyclomatic Complexity measure (henceforth referred to as "the complexity measure") is to limit the number of independent paths in a process. Yourdon and Constantine [Ref. 11] present the idea that an acceptable guideline for the length of a process is that the Structured English syntax or decision table be no more than one page in length. They acknowledge that this is a very general guideline but that it should be utilized in addition to ensuring that a process be strongly cohesive. However, as pointed out by McCabe, a 50 line process with 25 selection statements will result in 33.5 million control paths.

In order to determine what the complexity of a process should be, it must first be established how complex a process may be before a programmer can no longer effectively and rapidly understand it. Throughout managerial, psychological, and software engineering literature this complexity limit is known as the Hrair limit. As applied to software design, it has been determined to be seven logical events, plus or minus two logical events [Ref. 14]. The application of this limit to processes is that the number of basis paths through a process should be limited to seven. Such a limit will aid in testing and maintenance due to the ability of the maintenance personnel to review the design and quickly grasp the purpose of a process. This application has been

validated by a software reliability study [Ref. 16] which demonstrated that procedures of already coded software with 10 or more basis paths accounted for a much greater share of the errors. When processes are coded and compiled, their complexity will increase by 2 or 3; therefore, in the software design the complexity should be seven basis paths.

The theory behind the complexity metric is based on a definition and theorem from graph theory.

Definition 1. The cyclomatic number v(G) of a graph G with n vertices, e edges, and 1 connected component results in the equation:

$$v(G) = e - n + 1 \qquad (5.1)$$

Vertices are also known as nodes and an edge can be considered a path from one node to another.

Theorem 1. In a strongly connected graph, the cyclomatic number is equal to the maximum number of linearly independent paths.

A strongly connected graph is one in which there is a single entry point and a single exit point. All paths go from the entry point to the exit point. Furthermore there is a path from the exit point to the entry point. A module can be considered to be represented by a strongly connected graph because there is a single entry point from the calling module and the module returns control to that entry point when it is through processing.

When a control graph is drawn to represent the flow of control through a module, there is usually no indication of the path from the ending point to the entry point. McCabe remarks that this edge does not have to be drawn in, but

that it may be acccunted for by modifying equation 5.1 resulting in:

$$v(G) = (e + 1) - n + 1 \qquad (5.2)$$

or

$$v(G) = e - n + 2 \qquad (5.3)$$

Application of Theorem 1 to Graph G in Figure 5.1 shows that v(G) is 5. This indicates that there is a basis set of

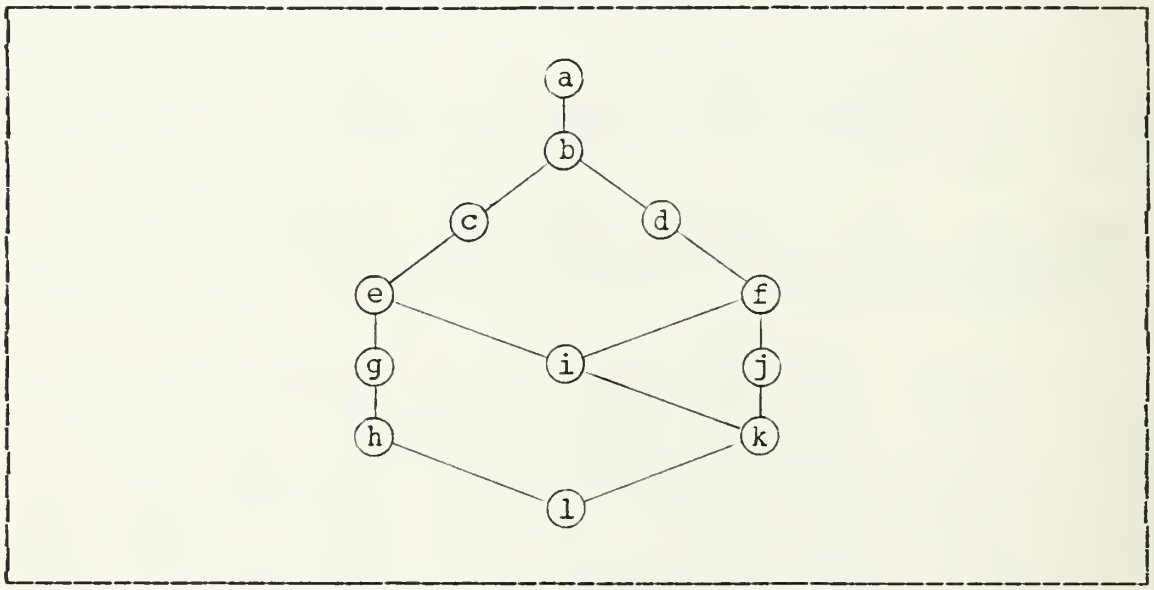

Figure 5.1 Graph G.

five independent paths from node 'a' to node 'l'. There is no one correct set of independent paths, but there is a basis of five paths. For example, there could be iterations of a loop within the module. The identification of the number of paths in the basis set does not tell how many iterations as the loop should be processed; that is

determined by data conditions at the decision points. Two examples of sets of basis paths for figure 5.1 are shown in table 6.

TABLE 6

Two sets of Basis Paths

	Set # 1	Set # 2
b1:	abcegheikl	abdfjkl
b2:	abdfikl	abceikl
b3:	abceikl	abdfikl
b4:	abceghl	abc(egh)^{3}eikl
b5:	abdfjkl	abc(egh)^{4}l

The notation $(egh)^3$ means to iterate the loop (egh) 3 times

B. PROCESS DESCRIPTIONS

1. Model Building

The purpose of these processes is to construct a format for new factors or groupings of factors. If it is a grouping of factors being constructed then identify the groups, the sub-groups, the group, sub-group, and factor coefficients (weights), each groups level within the model, and the sub-groups and factors of which they are composed. If there are new factors being formatted then obtain their factor identifiers, factor types, characteristics, and the subjective impact of the world and the factor itself upon the factor. If there are any factors which impact upon the factor being formatted then obtain their factor identifiers and their subjective impact upon the factor being formatted.

a. Scoring Model Construction

Get the Factors (or Factor) which the manager wishes to be part of a model or which will be forecast or analyzed at a later time.

Inputs:	MODEL_ID	Source:	Manager
	GROUP		Manager
	FACTOR_ID		Manager
	SUB_GROUP		Process 1.2
Outputs:	MODEL_STRUCTURE	Destination:	MODEL_STRUCTURE File
	FACTOR_ID		Process 1.2

b. Sub-Group Organization

Arrange Factors in Sub-Groups. Assign each Factor a weighting value and each Sub-Group a weighting value. Criteria for placing factors together in a Sub-Group are whether they are both either "desirable" or "undesirable" and that they can be traded off against one another. Otherwise they are in separate Sub-Groups. There is no limit to the number of Sub-Groups or Factors in a Sub-Group. However, single Factors do have to be assigned to a Sub-Group.

Inputs:	FACTOR_ID	Source:	Process 1.1
	SUB_GROUP_ID		Manager
	SUB_GROUP_WEIGHT		Manager
	FACTOR_WEIGHT		Manager
Outputs:	SUB_GROUP	Destination:	Process 1.1
	FACTOR_ID		Process 1.3

c. Addition to Factor File

If a Factor is a new Factor then get all information necessary for use in later analyzing or forecasting it.

Inputs:	FACTOR_ID	Source:	Process 1.2
	FACTOR		Manager
Outputs:	FACTOR	Destination:	FACTOR File

2. Model Management

The purpose of this process is to interpret the user command and forward the selection of the user for either analyzing or forecasting of the factor or model selected. Check to see if the selected factor or model is in the database. Any default values of the selection are set and the overall validity of the user's selection is verified. If it is not valid the user is notified of that fact and allowed to reenter a selection command.

a. Model Validation

Ensure that the user made a valid selection of options in his selection command.

Inputs:	USER_SELECT	Source:	Manager
	USER_SELECT		Process 6
	MODEL_STRUCTURE		Process 2.2
	FIRM_SELECT		Process 2.3
	VALIDATION		Process 2.2
Outputs:	FIRM_SELECT	Destination:	Process 4
	FIRM_SELECT		Process 6
	FACTOR_ID		Process 5
	FIRM_SELECT		Process 2.2

b. Set Default Values

Provide default values for any parameters not specified by the manager. These default values are a period of the forecast using data from 15 years prior to the present date to 15 years beyond the present date into the future. The default interval is one year. For the Monte Carlo selections the default number of iterations is 100.

Inputs:	USER_SELECT	Source:	Process 2.1
	TODAYS_DATE		Calendar
	MODEL_STRUCTURE		MODEL_STRUCTURE File
Outputs:	FIRM_SELECT	Destination:	Process 2.1

c. Ensure Sufficient Data Available

Check the Factor File to ensure that there are at least 3 data points within the user specified time period for each Factor necessary to the forecast. If the selection is for a cross-impact-analysis then this process is not necessary.

Inputs:	FIRM_SELECT	Source:	Process 2.1
	FACTOR		FACTOR File
	FACTOR_ID		Process 2.1
Outputs:	VALIDATION	Destination:	Process 2.1

3. Element Entry

This process allows operators to add values to the factors in the Factor file. It is a screen formatted entry and allows little leeway for the operator. Errors are possible if the operator enters the wrong units for the ELEMENT_VALUE in spite of the prompting by the process.

Inputs:	FACTOR_ID	Source:	Operator
	DATE_OF_OBSERVATION		Operator
	ELEMENT_SOURCE		Operator

ELEMENT_VALUE	Operator
DATE_OF_ENTRY	Calendar
UNITS	FACTOR File

Outputs: ELEMENT_ENTRY	Destination: FACTOR File
ENTRY_SCREEN	Operator

4. Forecast

This group of processes execute the forecast of the model or factor selected by the user. The forecast is made by fitting five types of curves to the observed data. The curve with the highest correlation coefficient is utilized for the forecast. A default confidence interval of 50% is applied for the forecast. The results of the forecast are provided to the manager in a tabular format. If individual factors are forecast then the results are placed in the Factor file, with an ELEMENT_ANALYSIS of "Estimate", ELEMENT_SOURCE is the CURVE used for the forecast, and DATE_OF_ENTRY and DATE_OF_OBSERVATION are the date and time the forecast was completed.

In the case of forecasting models three possible combinations are available. If all of the factors which make up the model have a factor limit then a model limit can then be calculated by utilizing the factor limit in place of the factor values in the scoring model and executing the model. This would enable Pearl and Gompertz curves to be calculated, because these curves require an upper limit to growth. The model can then be executed as normal with the model limit used in these two equations in place of the factor limit. If some factors in a model have no factor limit then only the linear, logarithmic, and double logarithmic curves are available for fitting.

There is also the option of forecasting each individual factor along the curve with the best fit and then

substituting the estimated values for each factor in the model. When observed data is not available then this is carried out through a Monte Carlo simulation using a random distribution between the upper and lower confidence limits of the estimate.

a. Limit Chcices

Determine the beginning and ending time of each interval of the user's request.

Inputs:	FIRM_SELECT	Source:	Process 2
	MODEL_STRUCTURE		MODEL_STRUCTURE File
	FACTOR_LIMIT		FACTOR File
Outputs:	BARE_FACTOR_VIEW	Destination:	Process 4.1.3
	BARE_FACTOR_VIEW		Process 4.1.2
	MODEL_STRUCTURE		Process 4.1.2
	MODEL_STRUCTURE		Process 4.1.4

b. Check For Model Limit

Check to see if Model-Limit can be calculated from data available. If all Factors in a model have a Factor-Limit then a Model-Limit can be calculated.

Inputs:	MODEL_STRUCTURE	Source:	Process 4.1.1
	FACTOR_LIMIT		Process 4.1.1
Outputs:	MODEL_LIMIT	Destination:	Process 4.1.4

c. Calculate the Model-Limit

Calculate the Model-Limit using the Factor-Limits for each Factor in the model and using the Model-Structure of the designated Model-Id.

Inputs:	MODEL_STRUCTURE	Source:	Process 4.1.2.1
	FACTOR_LIMIT		Process 4.1.2.1
Outputs:	MODEL_LIMIT	Destination:	Process 4.1.4

d. Screen Factor Values

Screen Factor File for historical data within an interval. Calculate the average of the values and note the number of values in each interval and the last interval which has any historical data in it.

Inputs:	BARE_FACTOR_VIEW	Source:	Process 4.1.1
	ELEMENT_VALUE		FACTOR File
Outputs:	FACTOR_VIEW	Destination:	Process 4.1.4
	FACTOR_VIEW		Process 4.3

e. Scoring Model

Ensure an interval has at least one data point in it prior to calculating the Model-Score. If there are no data points, go to the next interval.

Inputs:	MODEL_LIMIT	Source:	Process 4.1.2
	MODEL_STRUCTURE		Process 4.1.1
	FACTOR_VIEW		Process 4.1.3
	MODEL_SCORE		Process 4.1.4.2
Outputs:	MODEL_STRUCTURE	Destination:	Process 4.1.4.2
	AVG_VALUE		Process 4.1.4.2
	MODEL_VIEW		Process 4.3

f. Calculate Model-Score

Calculate score of a single interval of a model using the Model-Structure and the Avg-Values passed to it. The Factors which make up each Sub-Group are multiplied by their Factor-Weights and then summed together. All Sub-Groups are multiplied by their Sub-Group Weights. The Sub-Groups which are the components of each Group are multiplied together and then multiplied by the Group-Weights. Desirable Groups are divided by undesirable Groups.

Overriding Groups multiply the entire model.This product is the Mcdel-Score.

Inputs:	MODEL_STRUCTURE	Source:	Process 4.1.4.1
	AVG_VALUE		Process 4.1.4.1
Outputs:	MODEL_SCORE	Destination:	Process 4.1.4.1

g. Initialize Functions

Determine the set cf formulas which will be used to convert observed data into a linear form. The possible five curves are the Pearl curve, Gompertz curve, linear (no change), a logarithmic curve using the natural logarithm cf the dependent variable (the element-value), and a double-logarithmic curve using the natural logarithm of both the dependent (element- value) and independent variable (observation-date). If there are no Factor or Model Limits then the Pearl and Gcmpertz curves can not be utilized.

Inputs:	MODEL_VIEW	Source:	Process 4.1
	FACTOR_VIEW		Process 4.1
Outputs:	DATA_POINTS	Destination:	Process 4.3.1.3
	AVG_VALUE		Process 4.3.1.2
	END_INTERVAL		Process 4.3.1.2
	CURVE		Process 4.3.1.2
	LIMIT		Process 4.3.1.2

h. Calculate Curve Functions

Adjust the Avg-Values and Observation-Dates into a form which may be linear using the Gompertz, Pearl, Logarithmic, or Double-Logarithmic equations.

Inputs:	LIMIT	Source:	Process 4.3.1.1
	AVG_VALUE		Process 4.3.1.1
	END_INTERVAI		Process 4.3.1.1
	CURVE		Process 4.3.1.1

Outputs:	DEPENDENT	Destination:	Process 4.3.1.3
	INDEPENDENT		Process 4.3.1.3

i. Calculate Regression

Calculate A, B, the correlation coefficient, and the standard error of B through simple regression. The dependent variable (an adjusted or unadjusted Element-Value) is regressed on the independent variable (an adjusted or unadjusted Observation-Date).

Inputs:	DEPENDENT	Source:	Process 4.3.1.1
	DATA_POINTS		Process 4.3.1.1
	INDEPENDENT		Process 4.3.1.1
	LAST_OBSERVED_INTERVAL		Process 4.3.1.1

Outputs:	REGRESS_ANALYSIS	Destination:	Process 4.3.2
	REGRESS_ANALYSIS		Process 4.3.3
	REGRESS_ANALYSIS		Process 4.3.4
	REGRESS_ANALYSIS		Process 4.3.5
	REGRESS_ANALYSIS		Process 4.3.6

j. Pearl Curve Forecast

Generate estimates of data over the period in which data was observed using a Pearl curve formula, then calculate the estimated value for the data with an upper and lower limit for a 50% confidence interval. This information is provided for each interval from the end of observed data to the End-Period of the user request.

Inputs:	VARIATE	Source:	Process 4.3.1
	IDENTIFIER		Process 4.3.1
	REGRESS_ANAIYSIS		Process 4.3.1
	FACTOR_VIEW		Process 4.3.1
	MODEL_VIEW		Process 4.3.1

Outputs:	EXPANDED_MODEL_VIEW	Destination:	Manager
	EXPANDED_FACTOR_VIEW		Manager

EXPANDED_FACTOR_VIEW Process 4.3.1

k. Gompertz Curve Forecast

Generate estimates of data over the period in which data was observed using a Gompertz curve formula, then calculate the estimated value for the data with an upper and lower limit for a 50% confidence interval. This infcrmation is provided for each interval from the end of observed data to the End-Period of the user request.

Inputs:	VARIATE	Source:	Process 4.3.1
	IDENTIFIER		Process 4.3.1
	REGRESS_ANAIYSIS		Process 4.3.1
	FACTOR_VIEW		Process 4.3.1
	MODEL_VIEW		Process 4.3.1
Outputs:	EXPANDED_MCDEL_VIEW	Destination:	Manager
	EXPANDED_FACTOR_VIEW		Manager
	EXPANDED_FACTOR_VIEW		Process 4.3.1

l. Linear Curve Forecast

Generate estimates of data over the period in which data was observed using a Linear curve formula, then calculate the estimated value for the data with an upper and lower limit for a 50% confidence interval. This information is provided for each interval from the end of observed data to the End-Period of the user request.

Inputs:	VARIATE	Source:	Process 4.3.1
	IDENTIFIER		Process 4.3.1
	REGRESS_ANAIYSIS		Process 4.3.1
	FACTOR_VIEW		Process 4.3.1
	MODEL_VIEW		Process 4.3.1
Outputs:	EXPANDED_MCDEL_VIEW	Destination:	Manager
	EXPANDED_FACTOR_VIEW		Manager
	EXPANDED_FACTOR_VIEW		Process 4.3.1

m. Log Curve Forecast

Generate estimates of data over the period in which data was observed using a Logarithmic curve formula, then calculate the estimated value for the data with an upper and lower limit for a 50% confidence interval. This information is provided for each interval from the end of observed data to the End-Period of the user request.

Inputs:	VARIATE	Source:	Process 4.3.1
	IDENTIFIER		Process 4.3.1
	REGRESS_ANAIYSIS		Process 4.3.1
	FACTOR_VIEW		Process 4.3.1
	MODEL_VIEW		Process 4.3.1
Outputs:	EXPANDED_MCDEL_VIEW	Destination:	Manager
	EXPANDED_FACTOR_VIEW		Manager
	EXPANDED_FACTOR_VIEW		Process 4.3.1

n. Double-Log Curve Forecast

Generate estimates of data over the period in which data was observed using a Double-Logarithmic curve formula, then calculate the estimated value for the data with an upper and lower limit for a 50% confidence interval. This information is provided for each interval from the end of observed data to the End-Period of the user request.

Inputs:	VARIATE	Source:	Process 4.3.1
	IDENTIFIER		Process 4.3.1
	REGRESS_ANAIYSIS		Process 4.3.1
	FACTOR_VIEW		Process 4.3.1
	MODEL_VIEW		Process 4.3.1
Outputs:	EXPANDED_MCDEL_VIEW	Destination:	Manager
	EXPANDED_FACTOR_VIEW		Manager
	EXPANDED_FACTOR_VIEW		Process 4.3.1

o. Monte Carlo

Provide the actual Model-Score and the estimated Model-Score over the intervals with observed data. The estimate is arrived at through use of the estimated factor values for each Factor in the model substituted in a scoring model. For the intervals with no observed data a 50% confidence interval is established using the upper and lower estimates for each Factor of the model and substituting them into a scoring model. Then, for number of times specified by the user, a random value between the upper and lower estimates for each Factor is used for calculating the Model-Score.

Inputs:	FIRM_SELECT	Source:	Process 4.1
	EXPANDED_FACTOR_VIEW		Process 4.3
	MODEL_VIEW		Process 4.1
	MODEL_STRUCTURE		Process 4.1
	MODEL_SCORE		Process 4.4.2

Outputs:	MONTECARLO_FORECAST	Destination:	Manager
	MODEL_STRUCTURE		Process 4.4.2
	AVG_VALUE		Process 4.4.2

p. Calculate Model Score

Calculate score of a single interval of a model using the Model-Structure and the Avg-Values passed to it. The Factors which make up each Sub-Group are multiplied by their Factor-Weights and then summed together. All Sub-Groups are multiplied by their Sub-Group Weights. The Sub-Groups which are the components of each Group are multiplied together and then multiplied by the Group-Weights. Desirable Groups are divided by undesirable Groups. Overriding Groups multiply the entire model.This product is the Model-Score.

Inputs:	MODEL_STRUCTURE	Source:	Process 4.4.1

AVG_VALUE Process 4.4.1

Outputs: MODEL_SCORE Destination: Process 4.4.1

5. Cross Impact Analysis

This process utilizes the simple cross impact analysis model devised by Kane as discussed in [Ref. 2]. It searches the database for the values it requires and does not require any interaction by the user. Any changes tc the model will be made in the Model Management process.

a. Cross Impact

Construct a Cross-Impact-Matrix of Factors which impact on a single object Factor. This matrix also includes the impacts of the other Factors upon each other. The impact of the cutside world only impacts upon the Factors; the Factors do not impact upon the outside world.

Inputs:	FACTOR_ID	Source:	Process 2.0
	IMPACT_FACTCRS		FACTOR File
	FACTOR_IMPACTS		FACTOR File
Outputs:	CROSS_IMPACT_MATRIX	Destination:	Manager
	CROSS_IMPACT_MATRIX		Process 5.2

b. Calculate Impact

Over a relative period of time calculate the cumulative effects of the values of the Cross-Impact-Matrix upon a set of subjective Initial-Values provide by the manager for each Factcr.

Inputs:	CROSS_IMPACT_MATRIX	Source:	Process 5.1
	INITIAL_VALUE		Manager
Outputs:	RELATIVE_TIME	Destination:	Manager
	VALUE		Manager

6. Model Change

The user may change selected variables within the model which has most recently been executed and then execute it again. For a model forecast the Group-Weight, Sub-Group-Weight, and Factor-Weights may be changed. If desired, the modified model may be given a new name and placed in the Model-Structure file. The Factor-Limit of a Factor or of Factors in a model may be changed and the model run again. The selection parameters may be altered to change the time period looked at, the interval within the time period or, if the selection was for a model, a Monte Carlo forecast rather than a regular forecast (and vice versa) may be chosen.

Inputs:	INITIAL_VALUE	Source:	Process 5
	FACTOR_VIEW		Process 4
	MODEL_STRUCTURE		Process 4
	CROSS_IMPACT_MATRIX		Process 5
	FIRM_SELECT		Process 2
	GROUP_WEIGHT		Manager
	SUB_GROUP_WEIGHT		Manager
	FACTOR_WEIGHT		Manager
	FACTOR_LIMIT		Manager
	MODEL_ID		Manager

Outputs:	MODEL_STRUCTURE	Destination:	Process 4
	MODEL_STRUCTURE		MODEL_STRUCTURE File
	FACTOR_VIEW		Process 4
	CROSS_IMPACT_MATRIX		Process 5
	INITIAL_VAIUE		Process 5

VI. APPLICATIONS OF THE DSS

The ISDN concept is the integration of digital voice, circuit-switched data, and packet-switched data networks into a single network. The user of an ISDN would not be aware of which of these types of networks would be utilized to complete a connection to a destination. The network would select the required type of network, the choice of which would be transparent to the user, but is accessed at a single point. An ISDN architecture for the national backbone and distribution telecommunications systems could be desirable by the NCS as it would simplify the problem of integrating the present mix of communications networks in a national emergency.

The rate at which an ISDN architecture will evolve in the United States can not easily be mandated by regulation due to the increasing number of private companies and government agencies involved in construction and maintenance of telecommunications systems. As pointed out in Chapter 1, the impetus for growth of a technology comes from a need for that technology. In the United States, it is estimated that private users provide 85% of the needs driving the evolution of the common carrier network. Only 20% of the private users provide 80% of the use of the common carrier networks [Ref. 17]. Therefore, to forecast the growth of ISDN technology, it is necessary for the manager to forecast the need for an ISDN by private users.

The forecast DSS described in this paper can be utilized to forecast that user need for an ISDN. The method for doing this is for the NCS to collect data on the number of Private Branch Exchanges (PBX's) with digital capable microwave, optical fiber, or copper T-carrier direct tie-lines to toll

or tandem switches with direct access to the digital backbone system. This data can be expressed as a number of tie-lines installed or as a percentage of installed PBX's with the tie-lines. This is an indicator that network users want to bypass the local distribution systems which are difficult to use for high speed digital communications. The number of PBX's with tie-lines can be entered into the database of the DSS. A forecast can be generated of this data by extrapolating along the growth curve with the best fit to the data. This curve fitting is performed by the DSS and the results of the forecast are stored in the DSS database for later comparison and are also presented to the user in either tabular or graphical form.

The above example is one of an accelerator for technological growth of ISDN architectures. It would also be desirable to forecast constraints on the development and implementation of this technology. The number of engineers and maintenance personnel trained to install and maintain an ISDN is a definite constraint on implementation of an ISDN architecture in the United States. The data on the number of such ISDN trained personnel can be collected and forecast using the DSS.

After a number of accelerators and constraints have been collected, a cross impact analysis of the factors upon each other can be performed by the DSS. An analysis of this type could demonstrate the relative impact of different variables upon each other to obtain an approximation of the relative time required to achieve a desired result. For example, the impacts of digital communication media cost, the number of ISDN trained personnel, and the competition to provide digital services upon ISDN technology growth can be modeled. The DSS is executed with the impacts as subjective values and are defined as desirable or negative impacts upon ISDN technology.

An example cross impact matrix is given in Table 7. The variables listed in the table are:

1. Digital communications media cost = 'M'
2. Number of ISDN trained personnel = 'P'
3. Competition to provide digital services = 'C'
4. Growth rate of ISDN technology = 'G'

The cross impact matrix indicates that the cost of communications impacts negatively on the rate of growth, while the training of more personnel facilitates the training of even more personnel. The advantage of this model is that it demonstrates the relative impact that a combination of variables can have on the growth rates of other variables. Figure 6.1 is the result of executing the model with the values from the cross impact matrix of Table 7.

TABLE 7

Sample Cross Impact Matrix

Values of Impacting Factors

	M	P	C	G	Outside World
M	-.10	0.00	-.01	-.01	0.01
P	0.00	0.02	0.01	0.02	0.01
C	-.02	-.02	-.10	0.02	0.01
G	-.20	-.40	0.02	0.02	0.01

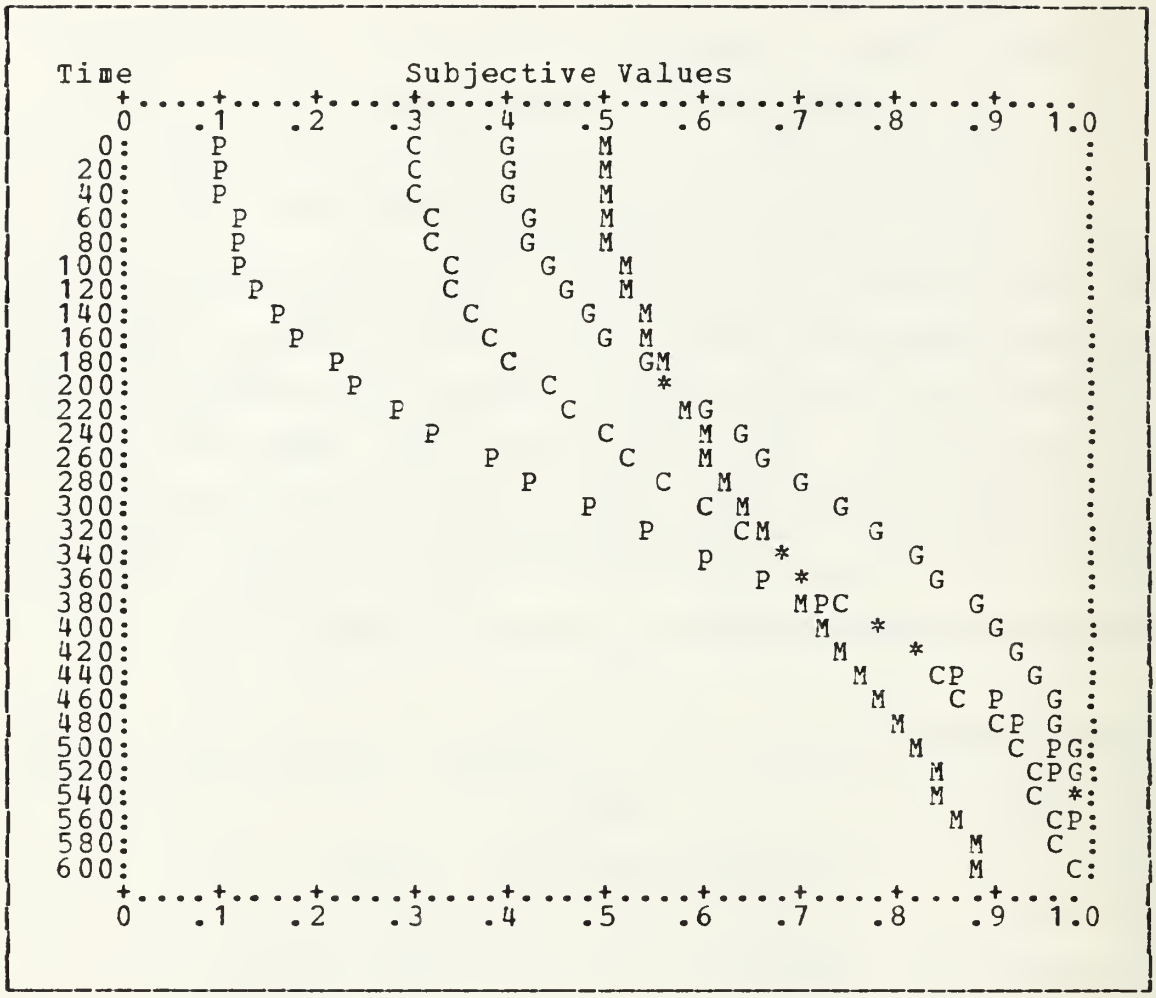

Figure 6.1 Output of Cross Impact Analysis.

VII. CONCLUSIONS

The National Communications System has been tasked with the overall responsibility for planning a national security/ emergency preparedness telecommunications system [Ref. 17]. An automated decisicn support system which can aid NCS managers in making effective forecasts of telecommunications technology, prices and costs would be an invaluable tool for the conduct of this planning. This work has described the logical design of such a system. The design is general enough to allow maximum flexibility in the eventual conversion to a physical, coded implementation. It will not be difficult to code this design in a higher order language such as Ada, COBOL, or BASIC. More importantly, the logical design cf this DSS lends itself toward implementation utilizing a fourth generation language such as FOCUS or NOMAD. The ability of these type of packages to access data through a database- type format allows a non-programmer to take this design and create a database which can be accessed by simple routines written in the generic language which accompanies these packages.

Furthermore, the design of this forecast DSS can be implemented on any size computer from a desktop microcomputer up to a large mainframe. The designation cf a specific system to run this package at this stage of the design is not necessary nor is it desirable. The end product that a user should be looking for is an acceptable logical design, not an up and running system. With a logical design the user can change ccmputer systems without having to have all of his software redesigned. It will be simple to have it coded for the new system or implement it himself with a fourth generation package as described earlier.

The modular design of this system enables expansion of the forecast routines with little effort. The models selected for this design were chosen for their simplicity and ease of understanding by the user. A complex econometric model may be more accurate (though that has been debated by professionals in the field [Ref. 18]), but the simplicity of simple regression models is more intuitive to the manager. Two possible simple models could be added, an exponential smoothing model, and a moving average model. However the seasonal or normalizing techniques which accompany these models are not so simple and depend on many more assumptions than a simple regression extrapolation. Through the use of the scoring model construction technique, the manager can build simple multivariable bootstrap models. Such models have been shown to model reality to a remarkable degree [Ref. 18]. In any case, this system's strict adherence to structured techniques and modularity would facilitate any future modification or expansion brought about by the dynamic nature of the telecommunications environment and the possibility of changes in overall system requirements.

APPENDIX A
ESSENTIALS OF STRUCTURED DESIGN

Stevens et al. [Ref. 19] introduced the concept of structured design as a comprehensive method for reducing the growing complexity of program design. Because it is not a true methodology, it is used most effectively in consonance with other techniques, such as structured programming, structured analysis, and HIPO hierarchy charts. The key to structured design is reducing the logical view of the system into simple pieces, called modules, that can be readily understood and hence constructed. The rationale behind this concept, common with most modern software design techniques, lies with principles of behavioral science regarding the human ability to comprehend and solve problems faster when they are of manageable size and complexity. These principles were the basis for top-down design, which calls for decomposing large, complex problems into smaller, less complex problems, until the original problem has been expressed as a combination of many, small, solvable problems. However, top-down design alone is not sufficient for ensuring modules that are easy to maintain and modify. Structured design includes the concept of top-down design, along with other strategies and heuristics. Among these are coupling and cohesion.

Coupling is a measure of the strength of association between separate modules within a system. The greater the degree of coupling, the harder it is to understand, change, or correct a module and hence the more complex and complicated the system. One goal of structured design is to create modules with coupling as weak as possible. This can be achieved, at least in theory, by designing the module

interface to be simple and obvious and ensuring the connection between modules is to the normal module interface (the entry point) rather than to module internal components.

Modules share a common environment when they interface with the same storage area, data region or device. Common environments often provide complex paths along which errors can travel when a change is made to one module. When common environments are originally designed into a system, add-on modules will also be forced to interface via the common environment, further degrading the product. Limiting common environment access to the smallest possible subset of modules tends to minimize this potential problem. Another method to achieve low coupling is to restrict interface connections to obvious relationships and avoid those that are inferred. Thus, connections which refer to a module as a whole require less coupling than those which refer to an internal component of a module. This latter case is called pathological connection, and is one of the strongest forms of coupling between modules. It can be avoided by ensuring a subroutine executes only when it is called formally by a module, it operates strictly on data passed by the calling module, only that data essential to the performance of its task is passed to the subroutine, and all results of its operations are returned to the calling module.

Cohesion is defined as the strength of association of the elements within a module, and is measured by a term called binding. The goal is to strive for high binding, which directly results in reduced coupling by minimizing the relationships among modules. The levels of cohesion may be addressed separately, scaled from low to high, and although a module may exhibit multiple levels of binding, the highest that may be applied determines the module level.

1. Coincidental binding means there is no meaningful relationship among the internal elements of a module.

It usually stems from haphazard attempts at breaking code up into "modules" or consolidating duplicate coding from several modules into one "module".

2. Logical binding means there exists some kind of logical relationship among the elements of a module. It often results from "cute", difficult to modify, shared code, or from passing unnecessary arguments.
3. Temporal binding means the elements are logically related also in time. Such elements are executed during a common period of time. The reason such modules are higher on the cohesion scale is that all the elements are at least executed at once.
4. Communicational binding means that elements are related further to the same input/output data set.
5. Sequential binding means that elements within a module are processed sequentially. It usually results from literal transformation of flowchart procedural blocks to modules. However, procedural processes can encompass more than one function.
6. Functional binding means all the elements of a module are related to the performance of a single function. It is the strongest level of binding. In practice, the determination of what exactly constitutes a function is a difficult task, further compounded by the dilemma of deciding how far to divide functionally bound subfunctions.

Although there may well be a basic tradeoff to be confronted between "structural design" modules and execution/memory overhead, there are a number of reasons why a structured design may, indeed, enhance execution time/memory space required. The major reasons are:

1. Error modules (called "optional") may never be called from memory.

2. Other, well-designed modules may only be executed a minimum number of times.
3. Structured design reduces the amount of duplicate or redundant code.

The structured design process is divided into two phases: general program design and detailed design. General program design is described as deciding what the program functions will be, and detailed design as deciding how the functions will be implemented. The overall design goal remains the structure of functionally bound, simply connected modules. The technique is simply top-down, modular, hierarchical with a unique graphical format. The following guidelines may be helpful when utilizing the structured design process:

1. In order to enhance maintainability, ensure the structure of the design matches the structure of the problem. Subsequent changes to the problem will then affect a minimal number of modules.
2. Strive for simple designs where the scope of effect of a decision is restricted to the scope of control of the module containing the decision. This is accomplished by either moving the decision element up in the structure chart, or by moving the entire module containing the decision so that it falls within the scope of control.
3. Use module size as an indicator of potential problems. A module that is extremely small may not perform a complete function. A module that is extremely large may include more than one function.
4. It is acceptable to design modules that return binary error or end-of-file flags. However, the same module should not be concerned with error recovery.
5. Duplicate code may, under certain circumstances, be acceptable. Duplicate functions, however, should be eliminated.

6. Particular data structures should be isolated in a minimum number of modules. This will facilitate module changes due to subsequent alterations in that data structure's specifications.
7. Minimize the number of parameters passed between modules. The goal is to pass only that data required by the module to accomplish its function.

There are several important variations of the basic structured design methodology. DeMarco [Ref. 12] proposes an approach that begins with the codification of the functional specification, or translating the prose specification into working fixed-format documents (data flow diagrams, data dictionary, transform descriptions, data structure chart). This step could actually be considered "structured analysis". The next step is the derivation of the structure chart, a modular hierarchy chart which records major design decisions and philosophy. Structure charts are recommended rather than flowcharts because flowcharts violate the principle of information hiding by exposing critical design decisions too early in the design process (for example, in what order and under what conditions functions are performed). Additionally, structure charts depict module connections and calling parameters, are smaller in size and generally more manageable. In the DeMarco version module design occurs next through construction of module descriptions. The final step is packaging the design, or shaping the logical design to accommodate the physical environment (machine, operating system, coding language, memory limitations, time restrictions). The key is to construct an environment-independent design first, maximizing cohesion and minimizing coupling, then impose packaging constraints so as to minimize degradation of product quality. An important structured design principle is to delay packaging as long as possible in order to "hide" the significant nature

of the design problem, to include algorithms, data structures and other transformations.

Enforcement of structured design techniques should significantly reduce the effort required for program modification and maintenance, if modules possess weak coupling and strong cohesion. Similarly, modules may be programmed, tested, and even optimized independently using these techniques. Structured design should, as a minimum, provide for "predictable" modules. These are modules which perform identically and consistently each time they are called, given identical inputs. Predictable modules also tend to perform independently of their environment. It is not clear, however, that strict adherence to structured design will ultimately result in a "library" of generalized, application-independent modules that may be easily configured to implement any sophisticated, complex system. In the final analysis, structured design is a method, not a methodology, and is to be used with other methods and tools to facilitate the design of programs.

APPENDIX B

PROCESS MINI-SPECIFICATIONS

```
**************************************************************
1.1 SCORING MODEL CONSTRUCTION

Inputs:  MODEL_ID              Source:  Manager
         GROUP                          Manager
         FACTOR_ID                      Manager
         SUB_GROUP                      Process 1.2

Outputs: MODEL_STRUCTURE       Destination: MODEL_STRUCTURE
                                              File
         FACTOR_ID                          Process 1.2

--------------------------------------------------------------

A.  Identify FACTOR_IDs that relate to how well the
      technology performs.
B.  Eliminate overlays of ELEMENTs from the model that
      measure the same or very similar characteristics.
D.  For each FACTOR_ID in the model
E.     Do SUB_GROUP_ORGANIZATION
F.  End For.
G.  For each SUB_GROUP_ID
H.     If each SUB_GROUP is of such an overriding nature that
         it must be present or the score of the model equals
         zero then
I.        Assign this SUB_GROUP to be the sole member of a GROUP
J.        Assign this GROUP a GROUP_ID
K.        Assign a GROUP_LEVEL = 0
L.        Assign a GROUP_TYPE = "Override"
       Else
M.        If SUB_GROUP is desirable then
N.           Assign to a GROUP with GROUP_TYPE = "Desirable"
          Else
O.           Assign to a GROUP with GROUP_TYPE = "Undesirable".
P.        End If
Q.     End If
R.  End For
S.  For each GROUP
T.     If GROUP_TYPE is not equal to "Override" then
U.        Assign a GROUP_ID.
V.        Assign a GROUP_LEVEL = 1.
W.     End If
X.  End For
Y.  Assign each GROUP a subjective GROUP_WEIGHT.
Z.  Assign the model a MODEL_ID.

--------------------------------------------------------------
```

TABLE 8

Process 1.1 Basis Paths

Complexity Metric: 7

1 abcfgrsxyz
2 abcdefgrsxyz
3 abcfghijklqrsxyz
4 abcfghmnpqrsxyz
5 abcfghmopqrsxyz
6 abcfghmopqrstwxyz
7 abcfghmopqrstuvwxyz

****

1.2 SUB_GROUP ORGANIZATION

Inputs:	FACTOR_ID	Source:	Process 1.1
	SUB_GROUP_ID		Manager
	SUB_GROUP_WEIGHT		Manager
	FACTOR_WEIGHT		Manager
Outputs:	SUB_GROUP	Destination:	Process 1.1
	FACTOR_ID		Process 1.3

A. Do ADDITION TO FACTOR File
B. If FACTORs can be traded off against FACTORs in
   a SUB_GROUP then
C. Assign FACTOR to that same SUB_GROUP.
D. Assign a FACTOR_WEIGHT
   Else
E. Assign FACTOR as sole member of a new SUB_GROUP.
F. Assign a FACTOR_WEIGHT
G. Assign a SUB_GROUP_ID.
H. Assign a subjective SUB_GROUP_WEIGHT.
I. End If

TABLE 9

Process 1.2 Basis Paths

Complexity Metric: 2

1. abcdi
2. abefghi

```
*************************************************************
1.3 ADDITION TO FACTOR FILE

Inputs: FACTOR_ID     Source: Process 1.2
        FACTOR                Manager

Outputs: FACTOR               Destination: FACTOR File
-------------------------------------------------------------

A. Check the FACTOR_ID against the FACTOR File
B. If it is not present then
C.     Get the FACTOR_ID along with the FACTOR_TYPE
           and CHARACTERISTIC from Manager
D.     If the FACTOR_TYPE = "Objective" then
E.        Get the UNITS and the FACTOR_LIMIT from Manager
F.     End If
G. End If
H. Get FACTOR_IMPACT of the FACTOR upon itself
I. Get FACTOR_IMPACT of the OUTSIDE_WORLD upon the FACTOR
J. If there are FACTORs which impact on this FACTOR then
K.     Provide the FACTOR_IDs
L.     Do ADDITION TO FACTOR FILE
M.     If these impacting FACTOR_IDs are not already
           listed in the FACTOR File as impacting on the
           object FACTOR then
N.        Get the subjective FACTOR_IMPACT
O.     End If
P. End If
-------------------------------------------------------------
```

TABLE 10

Process 1.3 Basis Paths

Complexity Metric: 5

1. abghijp
2. abcdefghijp
3. abcdfghijp
4. abghijklmnop
5. abghijklmop

****

2.1 MODEL VALIDATION

Inputs:	USER_SELECT	Source:	Manager
	USER_SELECT		Process 6
	MODEL_STRUCTURE		Process 2.2
	FIRM_SELECT		Process 2.2
	VALIDATION		Process 2.3
Outputs:	FIRM_SELECT	Destination:	Process 4
	FIRM_SELECT		Process 6
	FIRM_SELECT		Process 2.2
	FACTOR_ID		Process 5

```
A.   If a FACTOR_ID is in the USER_SELECT and CHOICE equals
       "Forecast" then
B.     If FACTOR_ID is present in the FACTOR File then
C.       Do SET_DEFAULTS
D.       Do ENSURE SUFFICIENT DATA AVAILABLE
E.     End If.
     Else
F.     If a MODEL_ID is in the USER_SELECT then
G.       Do SET DEFAULTS
H.       If CHOICE = "Cross Impact" then
I.         VALIDATION = "Not Valid"
J.       End If.
K.       If VALIDATION = "Valid" then
L.         Get the FACTOR_IDs from the MODEL_STRUCTURE
M.         For each FACTOR_ID
N.           Do ENSURE SUFFICIENT DATA AVAILABLE
O.         End For
P.       End If.
Q.     End If.
R.   End If.
```

TABLE 11

Process 2.1 Basis Paths

Complexity Metric: 7

1. aber
2. abcder
3. afqr
4. afghjkpqr
5. afghijkpqr
6. afghjklmnopqr
7. afghjklmopqr

```
**************************************************************
2.2 SET DEFAULT VALUES

Inputs: USER_SELECT              Source:  Process 2.1
        TODAYS_DATE                       Calendar
        MODEL_STRUCTURE                   MODEL_STRUCTURE File

Outputs: FIRM_SELECT             Destination:  Process 2.1
--------------------------------------------------------------

A. VALIDATION = "Valid"
B. If IDENTIFICATION = MODEL_ID and MODEL_ID is not in the
       MODEL_STRUCTURE File then
C.     VALIDATION = "Not Valid"
   Else
D.     Get TODAYS_DATE
E.     If BEGIN_PERIOD = Null then
F.        BEGIN_PERIOD = TODAYS_DATE - 15yrs
G.     End If.
H.     If END_PERIOD = Null then
I.        END_PERIOD = TODAYS_DATE + 15yrs
J.     End If.
K.     If BEGIN_PERIOD >= END_PERIOD then
L.        Selection is not valid
M.     End If.
N.     If INTERVAL = Null then
O.        INTERVAL = 1yr
P.     End If.
Q.     If CHOICE = "Monte Carlo" and ITERATIONS = Null then
R.        ITERATIONS = 100
S.     End If.
T. End If.

--------------------------------------------------------------
```

TABLE 12

Process 2.2 Basis Paths

Complexity Metric: 7

1. abct
2. abdeghjkmnpqst
3. abdefghjkmnpqst
4. abdeghijkmnpqst
5. abdeghjklmnpqst
6. abdeghjkmnopqst
7. abdeghjkmnpqrst

****

2.3 ENSURE SUFFICIENT DATA AVAILABLE

Inputs:	FIRM_SELECT	Source:	Process 2.1
	FACTOR		FACTOR File
	FACTOR_ID		Process 2.1
Outputs:	VALIDATION	Destination:	Process 2.1

A. If CHOICE = "Monte Carlo" or "Forecast" and at
least three ELEMENT_ENTRYs with an
ELEMENT_ANALYSIS equal to "Historical"
are NOT present with DATE_OF_OBSERVATION
between BEGIN_PERIOD and END_PERIOD then
B. VALIDATION = "Not Valid"
C. End If.

TABLE 13

Process 2.3 Basis Paths

Complexity Metric: 2

1. abc
2. ac

```
**********************************************************
3.0 ELEMENT ENTRY

Inputs: FACTOR_ID                Source: Operator
        DATE_OF_OBSERVATION              Operator
        ELEMENT_SOURCE                   Operator
        ELEMENT_VALUE                    Operator
        DATE_OF_ENTRY                    Calendar
        UNITS                            FACTOR File

Outputs: ELEMENT_ENTRY           Destination: FACTOR File
         ENTRY_SCREEN                       Operator

----------------------------------------------------------

A. If FACTOR_ID is in FACTOR File then
B.     Display ENTRY_SCREEN
C.     Enter DATE_OF_OBSERVATION
D.     Enter ELEMENT_SOURCE
E.     Enter ELEMENT_VALUE
F.     ELEMENT_ANALYSIS = "Historical"
G.     Combine with DATE_OF_ENTRY and store in FACTOR File
H. End If

----------------------------------------------------------
```

TABLE 14

Process 3.0 Basis Paths

Complexity Metric: 2

1. ah
2. abcdefgh

```
**************************************************************
4.1.1 LIMIT CHOICES

Inputs: FIRM_SELECT              Source: Process 2
        MODEL_STRUCTURE                  MODEL_STRUCTURE File
        FACTOR_LIMIT                     FACTOR File

Outputs: BARE_FACTOR_VIEW        Destination: Process 4.1.3
         BARE_FACTOR_VIEW                     Process 4.1.2
         MODEL_STRUCTURE                      Process 4.1.2
         MODEL_STRUCTURE                      Process 4.1.4

--------------------------------------------------------------

A.   INTERVAL_NUMBER = 0
B.   END_INTERVAL(0) = BEGIN_PERIOD
C.   While END_INTERVAL(INTERVAL_NUMBER) < END_PERIOD
D.      INTERVAL_NUMBER = INTERVAL_NUMBER + 1
E.      BEGIN_INTERVAL(INTERVAL_NUMBER) =
                    END_INTERVAL(INTERVAL_NUMBER - 1)
F.      END_INTERVAL(INTERVAL_NUMBER) =
                    BEGIN_INTERVAL(INTERVAL_NUMBER) + INTERVAL
G.   End While.
H.   LAST_INTERVAL = INTERVAL_NUMBER
I.   If MODEL_ID is in FIRM_SELECT and CHOICE = "Forecast"
        then
J.      For each FACTOR_ID in MODEL_STRUCTURE
K.         Do SCREEN FACTOR VALUES
L.      End For.
M.      Do CALCULATE MODEL LIMIT
N.      INTERVAL_NUMBER = 1
O.      While INTERVAL_NUMBER <= LAST_OBSERVED_INTERVAL
P.         Do SCORING MODEL
Q.         INTERVAL_NUMBER = INTERVAL_NUMBER + 1
R.      End While.
S.   End If.

--------------------------------------------------------------
```

TABLE 15

Process 4.1.1 Basis Paths

Complexity Metric: 5

1. abcdefghis
2. abcghis
3. abcghijlmnors
4. abcghijklmnors
5. abcghijlmnopqrs

```
**********************************************************
4.1.2.1  CHECK FOR MODEL LIMIT

Inputs: MODEL_STRUCTURE        Source: Process 4.1.1
        FACTOR_LIMIT                   Process 4.1.1

Outputs: MODEL_LIMIT           Destination: Process 4.1.4

----------------------------------------------------------

A.   For each FACTOR_ID in MODEL_STRUCTURE
B.      If FACTOR_LIMIT = Null then
C.         MODEL_LIMIT = Null
D.      End If.
E.   End For.
F.   If MODEL_LIMIT <> Null then
G.      Do CALCULATE MODEL_LIMIT
H.   End If.

----------------------------------------------------------
```

TABLE 16

Process 4.1.2.1 Basis Paths

Complexity Metric: 4

1. aefh
2. abdefh
3. abcdefh
4. abcdefgh

```
**********************************************************
4.1.2.2  CALCULATE MODEL LIMIT

Inputs: MODEL_STRUCTURE        Source: Process 4.1.2.1
        FACTOR_LIMIT                   Process 4.1.2.1

Outputs: MODEL_LIMIT           Destination: Process 4.1.4

----------------------------------------------------------

A.   For Each GROUP
B.      If GROUP_LEVEL = 1
C.         For each SUB_GROUP
D.            Sum the value of the FACTOR_LIMITs multiplied
                    by the FACTOR_WEIGHTS of each FACTOR_ID
E.            Multiply this sum by the SUB_GROUP_WEIGHT
F.            Remember this as the SUB_GROUP_VALUE
G.         End For
H.         Multiply the SUB_GROUP_VALUEs together
I.         Multiply the SUB_GROUP_VALUE by the
              GROUP_WEIGHT
J.         Remember this as the GROUP_VALUE
K.      End If.
L.   End For.
M.   If there are 2 groups with a GROUP_LEVEL = 1 then
N.      DIVIDE THE GROUP_VALUE of the GROUP which has a
           GROUP_TYPE = "Desirable" by the GROUP_VALUE
           of the GROUP which has GROUP_TYPE equal to
           "Undesirable"
O.      Remember this number as MODEL_LIMIT
P.   End If
Q.   For each GROUP
R.      If GROUP_LEVEL = 0
S.         GROUP_VALUE = FACTOR_LIMIT * GROUP_WEIGHT
T.         Multiply the GROUP_VALUE times the MODEL_LIMIT
U.         Remember this result as the MODEL_LIMIT
V.      End If.
W.   End For.

----------------------------------------------------------
```

TABLE 17

Process 4.1.2.2 Basis Paths

Complexity Metric: 7

1. almpqw
2. almpqrvw
3. abklmpqw
4. almpqrstuvw
5. abcghijklmpqw
6. abcdefghijklmpqw
7. almnopqw

```
**********************************************************************
4.1.3 SCREEN FACTOR VALUES

Inputs: BARE_FACTOR_VIEW      Source: Process 4.1.1
        ELEMENT_VALUE                 FACTOR File

Outputs: FACTOR_VIEW          Destination: Process 4.1.4
         FACTOR_VIEW                       Process 4.3

----------------------------------------------------------------------

A. For each INTERVAL_NUMBER
B.     If DATE_OF_OBSERVATION is greater than or equal
          to BEGIN_INTERVAL(INTERVAL_NUMBER) and less
          than END_INTERVAL(INTERVAL_NUMBER) then
C.       TOTAL = TOTAL + ELEMENT_VALUE
D.       OBSERVED = OBSERVED + 1
E.       LAST_OBSERVED_INTERVAL = INTERVAL_NUMBER
F.     End If.
G.     AVG_VALUE = TOTAL / OBSERVED
H. End For.

----------------------------------------------------------------------
```

TABLE 18

Process 4.1.3 Basis Paths

Complexity Metric: 3

1. abfgh
2. ah
3. abcdefgh

```
**************************************************************
4.1.4.1 SCORING MODEL

Inputs: MODEL_LIMIT              Source: Process 4.1.2
        MODEL_STRUCTURE                  Process 4.1.1
        FACTOR_VIEW                      Process 4.1.3
        MODEL_SCORE                      Process 4.1.4.2

Outputs: MODEL_STRUCTURE     Destination: Process 4.1.4.2
         AVG_VALUE                        Process 4.1.4.2
         MODEL_VIEW                       Process 4.3

--------------------------------------------------------------

A.   INTERVAL_NUMBER = 1
B.   While INTERVAL_NUMBER <= LAST_OBSERVED_INTERVAL
C.      GOOD_INTERVAL = 1
D.      For each FACTOR_ID in model
E.         If OBSERVED = 0 then
F.            GOOD_INTERVAL = 0
G.         End If.
H.      End For.
I.      If GOOD_INTERVAL = 1 then
J.         Do CALCULATE MODEL_SCORE
K.         OBSERVE = 1
        Else
L.         OBSERVE = 0
M.         MODEL_SCORE = 0
N.      End If.
O.      INTERVAL_NUMBER = INTERVAL_NUMBER + 1
P.   End While
Q. Combine data into MODEL_VIEW

--------------------------------------------------------------
```

TABLE 19

Process 4.1.4.1 Basis Paths

Complexity Metric: 5

1. alpq
2. abcdhilmnopq
3. abcdeghilmnopq
4. abcdefghilmnopq
5. abcdefghijknopq

```
**************************************************************
4.1.4.2  CALCULATE MODEL_SCORE

Inputs: MODEL_STRUCTURE       Source: Process 4.1.4.1
        AVG_VALUE                     Process 4.1.4.1

Outputs: MODEL_SCORE          Destination: Process 4.1.4.1

--------------------------------------------------------------

A.   For Each GROUP
B.      If GROUP_LEVEL = 1
C.         For each SUB_GROUP
D.            Sum the value of the AVG_VALUEs multiplied by
                 the FACTOR_WEIGHT of each FACTOR_ID
E.            Multiply this sum by the SUB_GROUP_WEIGHT
F.            Remember this as the SUB_GROUP_VALUE
G.         End For
H.         Multiply the SUB_GROUP_VALUEs together
I.         Multiply the SUB_GROUP_VALUE by the
              GROUP_WEIGHT
J.         Remember this as the GROUP_VALUE
K.      End If.
L.   End For.
M.   If there are 2 groups with a GROUP_LEVEL = 1 then
N.      DIVIDE THE GROUP_VALUE of the GROUP which has a
           GROUP_TYPE = "Desirable" by the GROUP_VALUE
           of the GROUP which has GROUP_TYPE equal to
           "Undesirable"
O.      Remember this number as MODEL_SCORE
P.   End If
Q.   For each GROUP
R.      If GROUP_LEVEL = 0
S.         GROUP_VALUE = AVG_VALUE * GROUP_WEIGHT
T.         Multiply the GROUP_VALUE times the MODEL_SCORE
U.         Remember this result as the MODEL_SCORE
V.      End If.
W.   End For.

--------------------------------------------------------------
```

TABLE 20

Process 4.1.4.2 Basis Paths

Complexity Metric: 7

1. almpqw
2. aklmpqw
3. abcghijklmpqw
4. abcdefghijklmpqw
5. almnopqw
6. almpqrvw
7. almpqrstuvw

```
**************************************************************
4.3.1.1 INITIALIZE FUNCTIONS

Inputs: MODEL_VIEW              Source: Process 4.1
        FACTOR_VIEW                     Process 4.1

Outputs: DATA_POINTS            Destination: Process 4.3.1.3
         AVG_VALUE                           Process 4.3.1.2
         END_INTERVAL                        Process 4.3.1.2
         CURVE                               Process 4.3.1.2
         LIMIT                               Process 4.3.1.2

--------------------------------------------------------------

A.   If FACTOR_LIMIT = Null then
B.      PICK = Set ['linear'|'Log'|'Double-Log']
     Else
C.      PICK = Set ['Pearl'|'Gompertz'|'Linear'|'Log'|'Double-Log']
D.      LIMIT = FACTOR_LIMIT
E.   End If.
F.   For CURVE = FIRST'LAST of set PICK
G.      I = 0
H.      COUNT = 1
I.      While COUNT <= LAST_OBSERVED_INTERVAL
J.         If OBSERVE(COUNT) > 0 then
K.            I = I + 1
L.            Do CURVE_FUNCTION
M.         End If
N.         COUNT = COUNT + 1
O.      End While.
P.      DATA_POINTS = I
Q.      Do CALCULATE REGRESSION
R.   End For
S.   Do SELECT CURVE

--------------------------------------------------------------
```

TABLE 21

Process 4.3.1.1 Basis Paths

Complexity Metric: 5

1. abefrs
2. acdefrs
3. abefghiopqrs
4. abefghijmnopqrs
5. abefghijklmnopqrs

```
**************************************************************
4.3.1.2 CALCULATE CURVE FUNCTIONS

Inputs: LIMIT                    Source: Process 4.3.1.1
        AVG_VALUE                        Process 4.3.1.1
        END_INTERVAL                     Process 4.3.1.1
        CURVE                            Process 4.3.1.1

Outputs: DEPENDENT             Destination: Process 4.3.1.3
         INDEPENDENT                        Process 4.3.1.3
--------------------------------------------------------------

A. Choose from the following:
      CURVE = Pearl
B.       DEPENDENT = LOG (LIMIT / AVG_VALUE - 1)
C.       INDEPENDENT = END_INTERVAL
      CURVE = Gompertz
D.       DEPENDENT = LOG (LOG (LIMIT / AVG_VALUE ) )
E.       INDEPENDENT = END_INTERVAL
      CURVE = Linear
F.       DEPENDENT = AVG_VALUE
G.       INDEPENDENT = END_INTERVAL
      CURVE = Logarithmic
H.       DEPENDENT = LOG (AVG_VALUE)
I.       INDEPENDENT = END_INTERVAL
      CURVE = Double_logarithmic
J.       DEPENDENT = LOG (AVG_VALUE)
K.       INDEPENDENT = LOG (END_INTERVAL)
L.  End Choice.
--------------------------------------------------------------
```

TABLE 22

Process 4.3.1.4 Basis Paths

Complexity Metric: 5

1. abcl
2. adel
3. afgl
4. ahil
5. ajkl

```
**************************  **********************************
4.3.1.3 CALCULATE REGRESSION

Inputs: DEPENDENT                 Source: Process 4.3.1.1
        DATA_POINTS                       Process 4.3.1.1
        INDEPENDENT                       Process 4.3.1.1
        LAST_OBSERVED_INTERVAL            Process 4.3.1.1

Outputs: REGRESS_ANALYSIS     Destination: Process 4.3.2
         REGRESS_ANALYSIS                  Process 4.3.3
         REGRESS_ANALYSIS                  Process 4.3.4
         REGRESS_ANALYSIS                  Process 4.3.5
         REGRESS_ANALYSIS                  Process 4.3.6

-------------------------------------------------------------

A.  Define Function R(Z) = A + B * Z
B.  For I = 1 to LAST_OBSERVED_INTERVAL
C.     P = P + DEPENDENT
D.     Q = Q + (DEPENDENT ** 2)
E.     R = R + INDEPENDENT
F.     S = S + (INDEPENDENT ** 2)
G.     U = U + (DEPENDENT * INDEPENDENT)
H.  End For.
I.  S1 = DATA_POINTS * S - R ** 2
J.  M2 = R / DATA_POINTS
K.  B = (DATA_POINTS * U - P * R) / S1
L.  A = (P - B * R) / DATA_POINTS
M.  V = E * SQR (S1 / (DATA_POINTS * Q - P ** 2))
N.  For I = 1 to DATA_POINTS
O.     N(I) = Fn R (INDEPENDENT)
P.     O(I) = DEPENDENT - N(I)
Q.  End For.
R.  For I = 1 to DATA_POINTS
S.     O1 = O1 + O(I) ** 2
T.  End For.
U.  O2 = O1 / (DATA_POINTS - 2)
V.  O3 = SQR (O2)
W.  B1 = (SQR (DATA_POINTS) * C3) / SQR (S1)
X.  VARIATE = 0.688 for a 50% confidence interval
Y.  Forecast CURVE with highest correlation factor => V

-------------------------------------------------------------
```

TABLE 23

Process 4.3.1.3 Basis Paths

Complexity Metric: 4

1. abhijklmnqrtuvwxy
2. abcdefghijklmnqrtuvwxy
3. abhijklmnopqrtuvwxy
4. abhijklmnqrstuvwxy

```
**************************************************************
4.3.2 PEARL CURVE FORECAST

Inputs: VARIATE                    Source: Process 4.3.1
        IDENTIFIER                         Process 4.3.1
        REGRESS_ANALYSIS                   Process 4.3.1
        FACTOR_VIEW                        Process 4.3.1
        MODEL_VIEW                         Process 4.3.1

Outputs: EXPANDED_MODEL_VIEW    Destination: Manager
         EXPANDED_FACTOR_VIEW                Manager
         EXPANDED_FACTOR_VIEW              Process 4.3.1

--------------------------------------------------------------

A.   Define Function R(Z) = A + B * Z
B.   Define Function P(Z) = LIMIT / (1 + EXP(Z))
C.   Define Function F(Z) = 03 * SQR(1 + 1 /
                           (DATA_POINTS + (DATA_POINTS
                            * (Z - M2) ** 2) / S1))
D.   Print 'A = ';EXP(A)
E.   Print 'B = ';-B
F.   Print 'Correlation Coefficient =';CORRELATION
G.   Print 'Standard Error of B =';B1
H.   INTERVAL = 1
I.   While INTERVAL <= LAST_OBSERVED_INTERVAL
J.      ESTIMATE = Function P( Function R( END_INTERVAL) ) )
K.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
L.      LCWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
M.      INTERVAL = INTERVAL + 1
N.   End While.
O.   INTERVAL = LAST_OBSERVED_INTERVAL + 1
P.   While INTERVAL <= LAST_INTERVAL_NUMBER
Q.      ESTIMATE = Function P( Function R( END_INTERVAL) ) )
R.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
S.      LCWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
T.      INTERVAL = INTERVAL + 1
U.   End While.

--------------------------------------------------------------
```

TABLE 24

Process 4.3.2 Basis Paths

Complexity Metric: 3

1. abcdefghinopu
2. abcdefghijklmnopu
3. abcdefghinopqrstu

```
**************************************************************
4.3.3 GOMPERTZ CURVE FORECAST

Inputs: VARIATE                    Source: Process 4.3.1
        IDENTIFIER                         Process 4.3.1
        REGRESS_ANALYSIS                   Process 4.3.1
        FACTOR_VIEW                        Process 4.3.1
        MODEL_VIEW                         Process 4.3.1

Outputs: EXPANDED_MODEL_VIEW   Destination: Manager
         EXPANDED_FACTOR_VIEW              Manager
         EXPANDED_FACTOR_VIEW              Process 4.3.1

--------------------------------------------------------------

A.   Define Function R(Z) = A + B * Z
B.   Define Function G(Z) = LIMIT * EXP(-EXP(Z))
C.   Define Function F(Z) = O3 * SQR(1 + 1 /
                           (DATA_POINTS + (DATA_POINTS
                            * (Z - M2) ** 2) / S1))
D.   Print 'B = ';EXP(A)
E.   Print 'K = ';-B
F.   Print 'Correlation Coefficient =';CORRELATION
G.   Print 'Standard Error of K =';B1
H.   INTERVAL = 1
I.   While INTERVAL <= LAST_OBSERVED_INTERVAL
J.      ESTIMATE = Function G( Function R( END_INTERVAL) ) )
K.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
L.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
M.      INTERVAL = INTERVAL + 1
N.   End While.
O.   INTERVAL = LAST_OBSERVED_INTERVAL + 1
P.   While INTERVAL <= LAST_INTERVAL_NUMBER
Q.      ESTIMATE = Function G( Function R( END_INTERVAL) ) )
R.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
S.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
T.      INTERVAL = INTERVAL + 1
U.   End While.

--------------------------------------------------------------
```

TABLE 25

Process 4.3.3 Basis Paths

Complexity Metric: 3

1. abcdefghinopu
2. abcdefghijklmnopu
3. abcdefghinopqrstu

```
**************************************************************
4.3.4 LINEAR CURVE FORECAST

Inputs: VARIATE                    Source: Process 4.3.1
        IDENTIFIER                         Process 4.3.1
        REGRESS_ANALYSIS                   Process 4.3.1
        FACTOR_VIEW                        Process 4.3.1
        MODEL_VIEW                         Process 4.3.1

Outputs: EXPANDED_MODEL_VIEW    Destination: Manager
         EXPANDED_FACTOR_VIEW                Manager
         EXPANDED_FACTCR_VIEW                Process 4.3.1

--------------------------------------------------------------

A.   Define Function R(Z) = A + B * Z
B.   Define Function F(Z) = O3 * SQR(1 + 1 /
                           (DATA_POINTS + (DATA_POINTS
                            * (Z - M2) ** 2) / S1))
C.   Print 'Intercept = ';A
D.   Print 'Slope = ';B
E.   Print 'Correlation Coefficient =';CORRELATION
F.   Print 'Standard Error of Slope =';B1
G.   INTERVAL = 1
H.   WHILE INTERVAL <= LAST_OBSERVED_INTERVAL
I.      ESTIMATE = Function R( END_INTERVAL) )
J.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
K.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
L.      INTERVAL = INTERVAL + 1
M.   End While.
N.   INTERVAL = LAST_OBSERVED_INTERVAL + 1
O.   While interval <= LAST_INTERVAL_NUMBER
P.      ESTIMATE = Function R( END_INTERVAL) )
Q.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
R.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
S.      INTERVAL = INTERVAL + 1
T.   End While.

--------------------------------------------------------------
```

TABLE 26

Process 4.3.4 Basis Paths

Complexity Metric: 3

1. abcdefghmnot
2. abcdefghijklmnct
3. abcdefghmnopqrst

```
**************************************************************
4.3.5 LOG CURVE FORECAST

Inputs: VARIATE                  Source: Process 4.3.1
        IDENTIFIER                       Process 4.3.1
        REGRESS_ANALYSIS                 Process 4.3.1
        FACTOR_VIEW                      Process 4.3.1
        MODEL_VIEW                       Process 4.3.1

Outputs: EXPANDED_MODEL_VIEW   Destination: Manager
         EXPANDED_FACTOR_VIEW              Manager
         EXPANDED_FACTOR_VIEW              Process 4.3.1

--------------------------------------------------------------

A.   Define Function R(Z) = A + B * Z
B.   Define Function F(Z) = O3 * SQR(1 + 1 /
                           (DATA_POINTS + (DATA_POINTS
                            * (Z - M2) ** 2) / S1))
C.   Print 'Constant Term = ';EXP(A)
D.   Print 'Growth Rate = ';B
E.   Print 'Correlation Coefficient =';CORRELATION
F.   Print 'Standard Error of Growth Rate =';B1
G.   INTERVAL = 1
H.   While INTERVAL <= LAST_OBSERVED_INTERVAL
I.      ESTIMATE = EXP( Function R( END_INTERVAL) ) )
J.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
K.      LCWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
L.      INTERVAL = INTERVAL + 1
M.   End While.
N.   INTERVAL = LAST_OBSERVED_INTERVAL + 1
C.   While INTERVAL <= LAST_INTERVAL_NUMBER
P.      ESTIMATE = EXP( Function R( END_INTERVAL) ) )
Q.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
R.      LCWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
S.      INTERVAL = INTERVAL + 1
T.   End While.

--------------------------------------------------------------
```

TABLE 27

Process 4.3.5 Basis Paths

Complexity Metric: 3

1. abcdefghmnot
2. abcdefghijklmnct
3. abcdefghmnopqrst

```
**************************************************************
4.3.6 DOUBLE_LOG CURVE FORECAST

Inputs: VARIATE                  Source: Process 4.3.1
        IDENTIFIER                       Process 4.3.1
        REGRESS_ANALYSIS                 Process 4.3.1
        FACTOR_VIEW                      Process 4.3.1
        MODEL_VIEW                       Process 4.3.1

Outputs: EXPANDED_MODEL_VIEW    Destination: Manager
         EXPANDED_FACTOR_VIEW                Manager
         EXPANDED_FACTOR_VIEW                Process 4.3.1

--------------------------------------------------------------

A.   Define Function R(Z) = A + B * Z
B.   Define Function F(Z) = 03 * SQR(1 + 1 /
                           (DATA_POINTS + (DATA_POINTS
                            * (Z - M2) ** 2) / S1))
C.   Print 'Constant = ';EXP(A)
D.   Print 'Power = ';B
E.   Print 'Correlation Coefficient =';CORRELATION
F.   Print 'Standard Error of Power =';B1
G.   INTERVAL = 1
H.   While INTERVAL <= LAST_OBSERVED_INTERVAL
I.      ESTIMATE = EXP( Function R( LOG( END_INTERVAL) ) )
J.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
K.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
L.      INTERVAL = INTERVAL + 1
M.   End While.
N.   INTERVAL = LAST_OBSERVED_INTERVAL + 1
O.   While INTERVAL <= LAST_INTERVAL_NUMBER
P.      ESTIMATE = EXP( Function R( LOG( END_INTERVAL) ) )
Q.      UPPER = ESTIMATE + ( VARIATE
                * Function F( END_INTERVAL) )
R.      LOWER = ESTIMATE - ( VARIATE
                * Function F( END_INTERVAL) )
S.      INTERVAL = INTERVAL + 1
T.   End While.

--------------------------------------------------------------
```

TABLE 28

Process 4.3.6 Basis Paths

Complexity Metric: 3

1. abcdefghmnot
2. abcdefghijklmnct
3. abcdefghmnopqrst

```
**************************************************************
4.4.1  MCNTE CARLO

Inputs: FIRM_SELECT              Source: Process 4.1
        EXPANDED_FACTCR_VIEW             Process 4.3
        MODEL_VIEW                       Process 4.1
        MODEL_STRUCTURE                  Process 4.1
        MODEL_SCORE                      Process 4.4.2

Outputs: MONTECARLO_FORECAST Destination: Manager
         MODEL_STRUCTURE                  Process 4.4.2
         AVG_VALUE                        Process 4.4.2

--------------------------------------------------------------

A.   For each FACTOR_ID in MODEL_STRUCTURE
B.      Do CURVE_FUNCTION
C.   End For
D.   INTERVAL_NUMBER = 1
E.   While INTERVAL_NUMBER <= LAST_OBSERVED_INTERVAL
F.      Do CALCULATE MODEL SCORE
G.      AVG_VALUE = ESTIMATE(FACTOR_ID)
H.      Do CALCULATE MODEL SCORE
I.      ESTIMATE(MODEL_ID) = MODEL_SCORE
J.      INTERVAL_NUMBER = INTERVAL_NUMBER + 1
K.      Print using OBSERVED_DATA_FORMAT
L.   End While.
M.   While INTERVAL_NUMBER <= LAST_INTERVAL_NUMBER
N.      AVG_VALUE = ESTIMATE(FACTOR_ID)
O.      Do CALCULATE MODEL SCORE
P.      ESTIMATE(MODEL_ID) = MODEL_SCORE
Q.      AVG_VALUE = UPPER(FACTOR_ID)
R.      Do CALCULATE MODEL SCORE
S.      UPPER(MODEL_ID) = MODEL_SCORE
T.      AVG_VALUE = LOWER(FACTOR_ID)
U.      Do CALCULATE MODEL SCORE
V.      LOWER(MODEL_ID) = MODEL_SCORE
W.      Print using ESTIMATED_DATA_FORMAT
X.      COUNT = 1
Y.      While COUNT <= ITERATIONS
Z.         For each FACTOR_ID in MODEL_STRUCTURE
A'.           AVG_VALUE = ((UPPER(FACTOR_ID)
                          - LOWER(FACTOR_ID))
                          * RANDOM_NUMBER) + LOWER(FACTOR_ID)
B'.        End For.
C'.        Do CALCULATE MODEL SCORE
D'.        FREQUENCY(INTERVAL_NUMBER,COUNT) = MODEL_SCORE
E'.        COUNT = COUNT + 1
F'.     End While.
G'.     INTERVAL_NUMBER = INTERVAL_NUMBER + 1
H'.     Print Frequency Distribution
I'.  End While.

--------------------------------------------------------------
```

TABLE 29

Process 4.4.1 Basis Paths

Complexity Metric: 6

1. acdelmi'
2. abcdelmi'
3. acdefghijklmi'
4. acdelmnopqrstuvwxyf'g'h'i'
5. acdelmnopqrstuvwxyzb'c'd'e'f'g'h'i'
6. acdelmnopqrstuvwxyza'b'c'd'e'f'g'h'i'

```
**************************************************************
4.4.2  CALCULATE MODEL_SCORE

Inputs: MODEL_STRUCTURE      Source: Process 4.4.1
        AVG_VALUE                    Process 4.4.1

Outputs: MODEL_SCORE         Destination: Process 4.4.1
--------------------------------------------------------------

A.   For Each GROUP
B.      If GROUP_LEVEL = 1
C.         For each SUB_GROUP
D.            Sum the value of the AVG_VALUEs multiplied by
                 the FACTOR_WEIGHTs for each FACTOR_ID
E.            Multiply this sum by the SUB_GROUP_WEIGHT
F.            Remember this as the SUB_GROUP_VALUE
G.         End For
H.         Multiply the SUB_GROUP_VALUEs together
I.         Multiply the SUB_GROUP_VALUE by the
              GROUP_WEIGHT
J.         Remember this as the GROUP_VALUE
K.      End If.
L.   End For.
M.   If there are 2 groups with a GROUP_LEVEL = 1 then
N.      DIVIDE THE GROUP_VALUE of the GROUP which has a
           GROUP_TYPE = "Desirable" by the GROUP_VALUE
           of the GROUP which has GROUP_TYPE equal to
           "Undesirable"
O.      Remember this number as MODEL_SCORE
P.   End If
Q.   For each GROUP
R.      If GROUP_LEVEL = 0
S.         GROUP_VALUE = AVG_VALUE * GROUP_WEIGHT
T.         Multiply the GROUP_VALUE times the MODEL_SCORE
U.         Remember this result as the MODEL_SCORE
V.      End If.
W.   End For.

--------------------------------------------------------------
```

TABLE 30

Process 4.4.2 Basis Paths

Complexity Metric: 7

1. almpqw
2. aɬklmpqw
3. aɬcghijklmpqw
4. aɬcdefghijklmpqw
5. almnopqw
6. almpqɾvw
7. almpqrstuvw

```
**************************************************************
5.1 CROSS IMPACT

Inputs: FACTOR_ID                Source: Process 2.0
        IMPACT_FACTORS                   FACTOR File
        FACTOR_IMPACTS                   FACTOR File

Outputs: CROSS_IMPACT_MATRIX Destination: Manager
         CROSS_IMPACT_MATRIX              Process 5.2
--------------------------------------------------------------

A. Identify the FACTOR_ID to observe
B. Get list of IMPACT_FACTORs
C. N1 = Number of IMPACT_FACTORs
D. For OBJECT = FIRST'LAST of set of IMPACT_FACTORs
E.     Get list of IMPACT_FACTORs for each OBJECT
F.     For IMPACT = FIRST'LAST of set of IMPACT_FACTORs
G.        If the IMPACT_FACTOR is not listed in the WORK
              File as an IMPACT_FACTOR on the OBJECT
              then
H.           CROSS_IMPACT_MATRIX( OBJECT,IMPACT ) = 0
          Else
I.           CROSS_IMPACT_MATRIX( OBJECT,IMPACT ) =
                 FACTOR_IMPACT
J.        End If
K.     End For.
L.     IMPACT = OUTSIDE_WORLD
M.     CROSS_IMPACT_MATRIX( OBJECT,IMPACT ) = WORLD_IMPACT
N. End For.

--------------------------------------------------------------
```

TABLE 31

Process 5.1 Basis Paths

Complexity Metric: 4

1. abcdn
2. abcdefklmn
3. abcdefgijklmn
4. abcdefghjklmn

```
**************************************************************
5.2 CALCULATE IMPACT

Inputs: CROSS_IMPACT_MATRIX  Source: Process 5.1
        INITIAL_VALUE                Manager

Outputs: RELATIVE_TIME       Destination: Manager
         VALUE                            Manager

--------------------------------------------------------------

A.   RELATIVE_TIME = 0
B.   For OBJECT= FIRST'LAST of set of IMPACT_FACTORs &
         OUTSIDE_WORLD
C.       Do CALCULATE INITIAL_VALUE
D.       VALUE(OBJECT) = INITIAL_VALUE
E.   End For
F.   TIME_INTERVAL = 0.001
G.   For RELATIVE_TIME= 1 to 1000
H.       For IMPACT = FIRST'LAST of set of IMPACT_FACTORs
I.           NEGATIVE(IMPACT) = DESIRABLE(IMPACT) = 0
J.           For OBJECT= FIRST'LAST of set of IMPACT_FACTORs &
                 OUTSIDE_WORLD
K.               NEGATIVE(IMPACT) = NEGATIVE(IMPACT)
                     + ((ABS(CROSS_IMPACT_MATRIX(IMPACT,OBJECT)))
                     - CROSS_IMPACT_MATRIX(IMPACT,OBJECT))
                     * VALUE(OBJECT)
L.               DESIRABLE(IMPACT) = DESIRABLE(IMPACT)
                     +((ABS(CROSS_IMPACT_MATRIX(IMPACT,OBJECT)))
                     + CROSS_IMPACT_MATRIX(IMPACT,OBJECT))
                     * VALUE(OBJECT)
M.           End For.
N.           E(IMPACT) = (1 + TIME_INTERVAL * (0.5)
                          * NEGATIVE(IMPACT)) /
                          (1 + TIME_INTERVAL * (0.5)
                          * DESIRABLE(IMPACT))
O.       End For.
P.       For IMPACT = FIRST'LAST of set of IMPACT_FACTORs
Q.           VALUE(IMPACT) = VALUE(IMPACT) ** E(IMPACT)
R.           If VALUE(IMPACT) <= 1.0 ** (-70)  then
S.               VALUE(IMPACT) = 0
T.           End If.
U.       End For.
V.   End For

--------------------------------------------------------------
```

TABLE 32

Process 5.2 Basis Paths

Complexity Metric: 7

1. abefgv
2. abcdefgv
3. abefghopuv
4. abefghijmnopuv
5. abefghijklmnopuv
6. abeghijmnopqrstuv
7. abeghijmnopqrtuv

```
**************************************************************
6 MODEL CHANGE

Inputs: INITIAL_VALUE          Source: Process 5
        FACTOR_VIEW                    Process 4
        MODEL_STRUCTURE                Process 4
        CROSS_IMPACT_MATRIX            Process 5
        FIRM_SELECT                    Process 2
        GROUP_WEIGHT                   Manager
        SUB_GROUP_WEIGHT               Manager
        FACTOR_WEIGHT                  Manager
        FACTOR_LIMIT                   Manager
        MODEL_ID                       Manager

Outputs: MODEL_STRUCTURE       Destination: Process 4
         MODEL_STRUCTURE                    MODEL_STRUCTURE
                                               File
         FACTOR_VIEW                        Process 4
         CROSS_IMPACT_MATRIX                Process 5
         INITIAL_VALUE                      Process 5

--------------------------------------------------------------

A.   Get FIRM_SELECT
B.   If IDENTIFIER is a MODEL_ID and CHANGE = "Model" then
C.      Change one of the following variables
D.         {GROUP_WEIGHT}
E.         {SUB_GROUP_WEIGHT}
F.         {FACTOR_WEIGHT}
G.   End If
H.   If CHANGE = "Factor" then
I.      For each FACTOR_ID
J.         Change FACTOR_LIMIT if desired
K.      End For.
L.   End If
M.   If CHANGE = "Selection" then
N.      Change one of the following variables
O.         CHOICE
P.         WINDOW
Q.         INTERVAL
R.      End Change.
S.   End If.

--------------------------------------------------------------
```

TABLE 33

Process 6 Basis Paths

Complexity Metric: 5

1. abghlms
2. abcdefghlms
3. abghijklms
4. abghiklms
5. abghlmnopqrs

APPENDIX C

DATA DICTIONARY

A. DATA FLOW COMPOSITIONS

```
BARE_FACTOR_VIEW      =  FACTOR_ID
                         (FACTOR_LIMIT)
                         WINDOW
                         INTERVAL
                         {INTERVAL_NUMBER
                          BEGIN_INTERVAL
                          END_INTERVAL}

BARE_MODEL_VIEW       =  MODEL_ID
                         (MODEL_LIMIT)
                         WINDOW
                         INTERVAL
                         {INTERVAL_NUMBER
                          BEGIN_INTERVAL
                          END_INTERVAL}

CROSS_IMPACT_MATRIX   =  FACTOR_ID
                         {IMPACT_FACTOR
                          FACTOR_IMPACT}

ELEMENT_ENTRY         =  DATE_OF_OBSERVATION
                         {ELEMENT_ANALYSIS
                          ELEMENT_SOURCE
                          DATE_OF_ENTRY
                          ELEMENT_VALUE}

EXPANDED_FACTOR_VIEW  =  FACTOR_ID
                         (FACTOR_LIMIT)
                         WINDOW
                         INTERVAL
                         LAST_OBSERVED_INTERVAL
                         LAST_INTERVAL_NUMBER
                         {INTERVAL_NUMBER
                          BEGIN_INTERVAL
                          END_INTERVAL
                          AVG_VALUE
                          (ESTIMATE
                           UPPER
                           LOWER)
                          OBSERVED}
```

```
EXPANDED_MODEL_VIEW   =   MODEL_ID
                          (MODEL_LIMIT)
                          WINDOW
                          INTERVAL
                          LAST_OBSERVED_INTERVAL
                          LAST_INTERVAL_NUMBER
                          {INTERVAL_NUMBER
                           BEGIN_INTERVAL
                           END_INTERVAL
                           MODEL_SCORE
                           (ESTIMATE
                            UPPER
                            LOWER)
                           OBSERVE}

FACTOR                =   FACTOR_ID
                          FACTOR_TYPE
                          CHARACTERISTIC
                          (UNITS + FACTOR_LIMIT)
                          {IMPACT_FACTOR
                           FACTOR_IMPACT}
                          OUTSIDE_WORLD
                          {ELEMENT_ENTRY}

FACTOR_VIEW           =   FACTOR_ID
                          (FACTOR_LIMIT)
                          WINDOW
                          INTERVAL
                          LAST_OBSERVED_INTERVAL
                          LAST_INTERVAL_NUMBER
                          {INTERVAL_NUMBER
                           BEGIN_INTERVAL
                           END_INTERVAL
                           AVG_VALUE
                           OBSERVED}

FIRM_SELECT           =   IDENTIFIER
                          CHOICE
                          VALIDATION
                          WINDOW
                          INTERVAL
                          (ITERATIONS)

IDENTIFIER            =   [MODEL_ID|FACTOR_ID]
```

```
MODEL_STRUCTURE      =  MODEL_ID
                        {GROUP_ID
                         GROUP_LEVEL
                         GROUP_WEIGHT
                         (GROUP_TYPE)
                         {SUB_GROUP}

MODEL_VIEW           =  MODEL_ID
                        (MODEL_LIMIT)
                        WINDOW
                        INTERVAL
                        LAST_OBSERVED_INTERVAL
                        LAST_INTERVAL_NUMBER
                        {INTERVAL_NUMBER
                         BEGIN_INTERVAL
                         END_INTERVAL
                         MODEL_SCORE
                         OBSERVE}

REGRESS_ANALYSIS     =  IDENTIFIER
                        CURVE
                        VARIATE
                        B
                        A
                        STANDARD_ERROR
                        CORRELATION
                        O3
                        M2
                        S1

SUB_GROUP            =  SUB_GROUP_ID
                        SUB_GROUP_WEIGHT
                        1{FACTOR_ID
                          FACTOR_WEIGHT}

USER_SELECT          =  IDENTIFIER
                        CHOICE
                        (WINDOW)
                        (INTERVAL)
                        (ITERATIONS)

WINDOW               =  BEGIN_PERIOD
                        END_PERIOD
```

B. DATA ELEMENT DESCRIPTIONS

```
A                    = type is digits 7
                       * Temporary variable in
                         regression calculation *

AVG_VALUE            = type is digits 7
                       * Average of all data elements in
                         a defined interval *
                       First defined in 4.0

B                    = type is FLOAT
                       * Temporary variable in
                         regression calculation *

BEGIN_INTERVAL       = type is range 0..100_000
                       * Julian date representation of
                         the date of the beginning of an
                         interval. Base year is 1900 *

BEGIN_PERIOD         = type is range 0..100_000
                       * Julian date representation of
                         the date of the beginning of
                         a period. Base year is 1900 *

CHARACTERISTIC       = type is (Endogenous,Exogenous)
                       * Identifies whether a FACTOR is
                         within users control or not *

CHOICE               = type is (Forecast, Monte-Carlo,
                         Cross-Matrix)
                       * Identify whether to run a
                         forecast model or the cross-
                         impact simulation *

CORRELATION          = type is digits 7 range 0.0..1.0
                       * This is the R ** 2 result of
                         regression analysis *
```

```
CURVE                    = type is (Pearl, Gompertz, Linear,
                             Log, Double-Log)
                           * Selection of curve function to
                             utilize *

DATE_OF_ENTRY            = type is range 0..100_000
                           * Julian date ELEMENT_ENTRY is
                             placed in the data base *

DATE_OF_OBSERVATION      = type is range 0..100_000
                           * Julian date ELEMENT_ENTRY's
                             ELEMENT_VALUE is observed *

ELEMENT_ANALYSIS         = type is (Historical,Estimate)
                           * Indicator of whether data is
                             Observed or a DSS generated
                             Estimate *

ELEMENT_SOURCE           = type is STRING(1..80)
                           * Source of data for
                             ELEMENT_ENTRY *

ELEMENT_VALUE            = type is digits 7
                           * Value of ELEMENT_ENTRY *

END_INTERVAL             = type is range 0..100_000
                           * Julian date representation of
                             the date of the ending of an
                             interval. Base year is 1900 *

END_PERICD               = type is range 0..100_000
                           * Julian date representation of
                             the date of the ending of
                             a period. Base year is 1900 *

ESTIMATE                 = type is digits 7
                           * An ELEMENT_VALUE generated by
                             the DSS *
```

```
FACTOR_ID          = type is STRING(1..21)
                     * Unique name of a FACTOR in the
                       format 'F.XXXXXXXXXX.TTTTTTTT'.
                       The 'F.' indicates it is a
                       FACTOR_ID, the 'X' is for the
                       name within a technology, the
                       'T' is for the technology *

FACTOR_IMPACT      = type is STRING(1.21)
                     * The FACTOR_ID of a FACTOR which
                       has an impact on the key
                       FACTOR *

FACTOR_LIMIT       = type is digits 7
                     * Highest value which an ELEMENT_
                       VALUE may ever be. Could be a
                       null value if there is no
                       limit *

FACTOR_TYPE        = type is (Subjective,Objective)
                     * Indicator of whether a FACTOR
                       is a subjective or objective
                       value *

FACTOR_WEIGHT      = type is delta 0.1 range 0.0..1.0
                     * a subjective weighting of a
                       FACTORs impact in a model *

GROUP_ID           = type is STRING(1..21)
                     * Unique name of a GROUP in the
                       format 'G.XXXXXXXXXX.TTTTTTTT'.
                       The 'G.' indicates it is a
                       GROUP_ID, the 'X' is for the
                       name within a technology, the
                       'T' is for the technology *
```

```
GROUP_LEVEL          = type is range 0..1
                       * Indicator of which GROUPs act
                         upon other GROUPs. Low numbers
                         act on high numbers *

GROUP_TYPE           = type is (Desirable,Undesirable)
                       * Indicator of whether a GROUP is
                         a desirable value or an
                         undesirable value *

GROUP_WEIGHT         = type is delta 0.1 range 0.0..1.0
                       * a subjective weighting of a
                         GROUPs impact in a model *

IMPACT_FACTOR        = type is delta 0.1 range 0.0..1.0
                       * Subjective value of the impact
                         of one FACTOR upon another *

INTERVAL             = type is range 1..100_000
                       * Length of each interval over
                         which to average data values
                         for forecasting as measured in
                         days *

INTERVAL_NUMBER      = type is range 1..400
                       * Number of intervals in the
                         WINDOW defined by the user *

ITERATIONS           = type is range 1..500
                       * Number of types to execute
                         Monte Carlo simulation *
```

```
LAST_INTERVAL_NUMBER     =  type is range 1..400
                            * Last INTERVAL_NUMBER in WINDOW
                              defined by the user *

LAST_OBSERVED_INTERVAL   =  type is range 1..400
                            * Last interval which contains
                              data which is Historical *

LOWER                    =  type is digits 7
                            * The lower value of the
                              confidence limit for an
                              interval in regression
                              analysis *

MODEL_ID                 =  type is STRING(1..21)
                            * Unique name of a
                              MODEL_STRUCTURE in the
                              format 'M.XXXXXXXXXX.TTTTTTTT'.
                              The 'M.' indicates it is a
                              MODEL_ID, the 'X' is for the
                              name within a technology, the
                              'T' is for the technology *

M2                       =  type is digits 7
                            * Temporary variable in
                              regression calculation *

OBSERVED                 =  type is range 0..1000
                            * Number of ELEMENT_VALUES in an
                              INTERVAL *

OUTSIDE_WORLD            =  type is delta 0.1 range 0.0..1.0
                            * Subjective impact of world upon
                              a FACTOR *

C3                       =  type is digits 7
                            * Temporary variable in
                              regression calculation *
```

```
RELATIVE_TIME          = type is range 0..1000
                         * An counter of relative time
                           periods in the Cross Impact
                           Analysis *

STANDARD_ERROR         = type is digits 7
                         * The standard error of the
                           estimate of the dependent
                           variable in the regression
                           analysis *

SUB_GROUP_ID           = type is STRING(1..21)
                         * Unique name of a SUB_GROUP in
                           format 'S.XXXXXXXXXX.TTTTTTTT'.
                           The 'S.' indicates it is a
                           SUB_GROUP_ID, the 'X' is for
                           name within a technology, the
                           'T' is for the technology *

SUB_GROUP_WEIGHT       = type is delta 0.1 range 0.0..1.0
                         * a subjective weighting of a
                           SUB_GROUPs impact in a model *

S1                     = type is digits 7
                         * Temporary variable in
                           regression calculation *

UNITS                  = type is STRING(1..20)
                         * Units of measure for
                           ELEMENT_VALUEs *
```

```
UPPER          =  type is digits 7
                  * The upper value of the
                    confidence limit for an
                    interval in regression
                    analysis *

VALIDATION     =  type is (Valid,Not-Valid)
                  * Indicator of whether a
                    USER_SELECT is acceptable *

VARIATE        =  type is delta 0.001
                          range 0.000..1.000
                  * Value to determine the
                    confidence interval in
                    analysis *
```

LIST OF REFERENCES

1. National Security Telecommunications Policy (PD/NSC-97), Office of the Manager, National Communications System, 3 August 1983.

2. Martino, Joseph P., Technological Forecasting for Decision Making, Elsevier Publishing Co., 1983.

3. Bright, James R., Technological Forecasting for Industry and Government: Methods and Applications, Prentice-Hall, Inc., 1968.

4. NCS Organization and Functions Manual, Office of the Manager, National Communications System, 22 May 1975.

5. National Security Telecommunications Policy (PD/NSC-53), Office of the Manager, National Communications System, 15 November 1979.

6. National Communications System Instruction 45-4, NCS Telecommunications Emergency Management System Organization and Operations Manual, January 1979.

7. NCS Management of Telecommunications Resources and Services During a Federal Emergency - Objectives and Provisional Guidelines, Office of the Manager, National Communications System, April 1980.

8. Cheslow, R.T. and J.R. Dever, "Acquisition Costing in the Federal Government", Defense Systems Management Review, vol 2, No. 4, 1979.

9. Intriligator, Michael D., Econometric Models, Techniques, and Applications, Prentice-Hall, Inc., 1978.

10. Naval Postgraduate School Report NPS-54-83-012, A Knowledge-Based System For LP Modeling, by Dolk, D.R., 10 January 1983.

11. Yourdon, Edward and Larry L. Constantine, Structured Design: Fundamentals of a Discipline of Computer Program and System Design, Yourdon Press, 1978.

12. DeMarco, Tom, Structured Analysis and System Specification, Yourdon Press, 1978.

13. Bohm, C. and G. Jacopini, "Flow Diagrams, Turing Machines and Languages With Only Two Formation Rules", Communications of the ACM, vol. 9, No. 5, 1966.

14. McCabe, Thomas J., Leon F. Young . Wayne Clayba n and Jim McManus, "Design Basis hs: A Complexity Driven Design Inspection Methodol ", IEEE 1983 Total Systems Reliability Symposium, December 1983.

15. Boehm, Barry W., R.K. McClean, and D.B. Urfrig, "Some Experience with Automated Aids to the Design of Large-Scale Reliable Software", IEEE Transactions on Software Engineering, vol SE-1, 1975.

16. Walsh, T.J., "A Software Reliability Study Using a Complexity Measure", Proceedings of the National Computer Conference, AFIPS, 1979.

17. SRI Internaticnal Technical Note SSC-TN-ISR-13, Framework For National Communications System (NCS) Strategic Planning, by Foster, B.F. and D.L. Miller, February 1981.

18. Armstrong, J. Scott, Long Range Forecasting: From Crystal Ball to Computer, John Wiley Inc., 1978.

19. Stevens, W.P., G.J. Myers and L.I. Constantine, "Structured Design", IBM Systems Journal, vol. 13, No. 2, 1974.

BIBLIOGRAPHY

Blum, H. and Karl Steinbeck, ed., Technological Forecasting in Practice, Heath Ltd., 1972.

Booth, Grayce M., The Design of Complex Information Systems, McGraw-Hill, 1983.

DeMarco, Tom, Controlling Software Projects, Yourdon Press, 1982.

Jussawall, M. and L.M. Lamberton, ed., Communication Economics and Development, Pergamon Press, 1982.

Keen, Peter G.W., Michael S. Scott Morton, Decision Support SystemsAn Organizational Perspective, Addison-Wesley Publishing Company, 1978.

Leventach, H. and J.P. Cleary, The Modern Forecasting Process Through Data Analysis, Lifetime Learning Publications, 1984.

Makridakis, Spyros and Steven C. Wheelwright, Forecasting Methods and Applications, John Wiley and Sons, 1978.

Martino, Joseph P., An Introduction to Technological Forecasting, Gordon and Breach, Science Publishers Inc., 1972.

Sharma, R.L., P.J.T. deSousa and A.D. Ingle, Network Systems, Von Nostrand Reinhold Co., 1982.

Sprague, Ralph H. Jr., Eric D. Carlson, Building Effective Decision Support Systems, Prentice-Hall, 1982.

Yourdon, Edward, Managing The Structured Techniques, Prentice-Hall, Inc., 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, Califcrnia 93943	2
3.	Computer Technology Curricular Office Code 37 Naval Postgraduate School Monterey, Califcrnia 93943	1
4.	IT Kent A. Williams 866 North Street Peekskill, NY 10566	1
5.	CPT Edwin C. Partridge III USAE AFCENT Box 215 APO NY 09011	1

210839

Thesis
W607 Williams
c.1 Logical design of a decision support system to forecast technology, prices and costs for the National Communications System.