A COMPUTER-BASED CORPORATE MODELING SYSTEM

By

ROBERT FRANKLIN ZANT

A DISSERTATION PRESENTED TO THE GRADUATE COUNCIL OF
THE UNIVERSITY OF FLORIDA
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA
1972

To Sue, Lynn, and Tina

ACKNOWLEDGMENTS

The writer wishes to express his gratitude to the members of his supervisory committee: Dr. William V. Wilmot, Jr., Chairman; Dr. R. H. Blodgett; Dr. E. L. Jackson; and Dr. W. E. Stone. They have generously contributed their time and efforts in assisting the writer in this study.

The writer is particularly indebted to Dr. Wilmot, who as a professor, department head, and supervisory committee chairman, has contributed immeasurably to the writer's graduate studies.

Mr. John Kisner, Mr. Joe Gothe, and Mr. Tom Cox of the Riegel Textile Corporation have also provided much-needed advice and encouragement throughout the course of this study.

Finally, the writer wishes to acknowledge the self-denial which was graciously accepted by his wife, Sue, and daughters, Lynn and Tina. Special appreciation is extended to his wife who typed all drafts and the final manuscript for this study.

TABLE OF CONTENTS

TABLE OF CONTENTS

(continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES

(continued)

Abstract of Dissertation Presented to the Graduate Council of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

A COMPUTER-BASED CORPORATE MODELING SYSTEM

By

Robert Franklin Zant

August, 1972

Chairman: Dr. William V. Wilmot, Jr.
Major Department: Department of Management and Business Law

The use of formalized computer-based models in the management of a firm has been primarily concentrated in operational areas. There is, however, a growing interest in the development of models which are broader in scope than operational models. Models which interrelate all areas of the firm - production, finance, and marketing - are called corporate models.

Past efforts in the area of corporate modeling have demonstrated the need for special purpose computer languages which facilitate the construction of corporate models. The purpose of this study is to determine the desirable characteristics of such languages and then to develop prototype languages which supply the facilities necessary to support a corporate modeling effort.

A modeling system is developed which is composed of three languages - a control language, a logic and data specification language, and a report format specification language. Some of the features of the new system are

1) a high level of user orientation; 2) the ability to perform sensitivity analysis; 3) the ability to perform Monte Carlo simulations; and, 4) the ability to transfer the values of variables from one model to another.

The high level of user orientation of the modeling system developed in this study makes the system particularly well suited for the initial modeling effort of a firm, for the development of exploratory and/or short-life models, and as a pedagogical aid in introducing managers to the concept of modeling.

CHAPTER I

INTRODUCTION

One of the fundamental concepts in management theory is the concept of models. Managers use models everyday in forming opinions and in making "intuitive" judgments as well as in making decisions "scientifically." Indeed, everyone interprets his perceptions in accordance with a conceptual or mental model known as the person's "view of the world." Many psychologists maintain that if a person's "view of the world" can be understood then his behavior can be predicted. And, a person's behavior can be altered by altering his conceptual model. The implication for management scientists is that they should be concerned with the formalization of the models which managers are currently using (a <u>positive</u> theory) and the models which managers should be using (a <u>normative</u> theory). Management theory would thus form a base for the improvement and the development of management techniques.

The formalization of models used by management has been mostly concentrated in operational areas such as production scheduling and inventory control systems. Operational problems are more easily formalized because they tend to be repetitive in nature and more well structured than other managerial problems. There is, however, a great deal of interest in the development of models which are broader in scope than operational models, models which consider the total corporation; i.e., corporate models.

Definition of Corporate Models

The activity in the area of corporate modeling has been so diverse that one author has concluded that there is "no generally accepted definition of a corporate model..."[11, p. 43]. Models reported in the literature have varied from marketing oriented models [12] to production oriented models [2] to models used only in investment analysis [20, 21]. In most cases, however, the models which are purported to be corporate models have the common trait of considering the impact of an action over the total organization. This is usually accomplished by considering the impact on the corporate financial statements; i.e., the income statement, the balance sheet, cash flow statement, etc.

The most trivial corporate model would simply be the relationship "corporate income equals total revenue minus total expenses." This simple model would be utilized by estimating the effect of a proposed action on revenues and on expenses and then computing the estimated corporate income. Of course, a model at this level of aggregation would be of little use. In order to make the model more meaningful, the detail of the model may be increased by segmenting the revenues and expenses by divisions and by products. The model would then consist of relationships such as "corporate income equals the sum of the divisions' net incomes minus corporate expenses" and "a division's net income equals the sum of the net contributions per product minus divisional expenses." The model could then be used to study the effect of alternatives such as the reallocation of advertising funds among products; or the effect of a new pricing policy; or, if the model were sufficiently detailed, the effect of a change in the production process. In this way the corporate model could be used to analyze the effects of various alternative actions on the entire corporation.

Functions of Corporate Models

The functions of corporate models have been defined in varying ways by several different authors [see 16; 9; 13; and 7]. However, the functions may be generally classified into three groups. First, corporate models serve as a focus for planning and decision-making efforts; second, they assist in the analysis of alternative courses of action; and, third, they assist in the coordination of the firm's planning and decision-making activities.

A corporate model serves as a focus for planning and decision-making efforts during the developmental stage of the model and serves as a continuing focus as the model is being used. The development of a corporate model requires the clear and precise specification of the inter-relationships between elements within the firm. Since the model is a representation of management's concept of the firm, the creation of the model will highlight the areas in which the manager's conceptual models are "fuzzy" or ill-defined. By focusing attention on these areas, the model encourages management to clarify their perceptions of the functional inter-relationships in the organization. The creation of the model thereby causes management to re-examine and to improve their understanding of the firm. The continual use of the model then both encourages and guides the re-evaluation and improvement of this understanding [16, p. 611].

The rephrasing of management's perceptions in the form of a model also has the effect of defining the information needed in the planning and decision-making process. "As a matter of fact, the approach used to develop a corporate model is very similar to the method used to develop the requirements for an information system, e.g., the identification of key variables" [9, p. 33]. A model would define, for example, the classification of costs or the form of information such as market forecasts which are needed for managerial planning and decision-making.

Perhaps the most obvious function of corporate models is that they may be used in the analysis of alternative courses of action. A corporate model represents a comprehensive and interrelated view of the firm. Consequently, the use of such a model broadens the scope of the analysis of an alternative both in terms of the organizational unit and the time span considered. That is, it is possible to investigate the long-term effects of an alternative on the entire organization. Furthermore, since corporate models are usually solved with little delay by use of a computer, more alternatives can be considered than would be otherwise possible. Of course, the shorter response time could be used to make decisions more rapidly instead of being used to allow the consideration of more alternatives. Or, a decision may be delayed until more information is available. In any event, the use of a corporate model has a significant impact on the managerial process.

The use of the computer in solving corporate models also has the advantage of improving the analysis of alternatives through improved accuracy, through the determination of critical variables, and through risk analysis.

Although the saying "Garbage In-Garbage Out" is certainly true with respect to computers, it is also true that, given correct information and a correct formulation of the procedure to be followed, the computer is much less prone to error than are humans. This increased accuracy is essential in a corporate modeling effort because of the number of manipulations and the magnitude of the data base which is involved.

The use of a corporate model in conjunction with sensitivity analysis aids in the determination of the critical variables for a particular decision. Sensitivity analysis is accomplished by repeatedly solving a model with the value of one input variable being changed between each solution. If the behavior of the model changes significantly for small changes in the input variable, then the system is said to be sensitive to the value of the input variable. By carrying out sensitivity analyses over many input variables, the critical variables may be determined. More effort can then be expended in estimating and controlling the critical variables.

Corporate models may also be used in analyzing the risk involved with the acceptance of various alterations. This is usually accomplished by the use of Monte Carlo simulation. Monte Carlo simulation refers to the process of sampling from probabilistic variables and then using the sample values in solving a mathematical model. The sampling-solution process is repeated numerous times in order to develop a distribution of the responses of the model. In the case of a corporate model the distribution would typically describe the profitability of an alternative. The distribution would depict the risk associated with the alternative by disclosing the probability of loss as well as the probability of receiving less than acceptable returns.

The third function of corporate models is the coordination of planning and decision-making activities. The development of a corporate model requires the explicit consideration of the assumptions that are to be made, the explicit statement of the inter-relationships that are to be considered, and the specification of the information which is needed. By unambiguously specifying the relationships among variables and the values of variables, a model represents a common view of the firm which all managers can comprehend. Planning and decision-making activities are thereby undertaken with all parties having a common frame of reference. As McKenney puts it, a model "is discernible to a variety of managers and therefore discussible" [16, p. 600].

Models also assist in coordination by virtue of their comprehensiveness. When an alternative is analyzed, its effect on the whole firm is considered. If a new policy is to be considered, "it is possible to study how it reverberates and affects the entire company" [9, p. 33].

The use of models can also improve the consistency of decision-making throughout the firm. The use of a model over time and the use of a model by different areas in an organization insures that the same variables are considered in each decision-making process. If a process and model are changed, then the new model would represent an explicit statement of the new assumptions and criteria that are to be used.

This broad use of modeling for consistency in decision-making is exemplified by the Monsanto Company. The Monsanto Company has developed a modeling language which affords them a convenient though advanced modeling capability. Monsanto now requires that all proposals for large capital projects contain the results of sensitivity and risk analysis [4, p. 12]. Proposals from all areas of the firm can now be compared on the basis of common economic assumptions.

In review, corporate models function as a focus for the initial activity in a planning effort and as a focus for the continued improvement of such efforts. Corporate models aid in the analysis of alternatives because the use of a computer allows the rapid analysis of complex models. Corporate models also assist in the determination of critical values through sensitivity analysis and the consideration of risk through Monte Carlo simulation. Finally, corporate models assist in the corrdination of planning and decision-making activities throughout the organization.

Developments in Corporate Modeling

The construction of formalized corporate models is a relatively new endeavor. The first modeling efforts began in the late 1950's, but widespread interest in corporate modeling did not develop until the late 1960's [9, p. 29]. The recent growth in corporate modeling seems to be a result of the fact that the initiation of a corporate modeling effort is usually an extension of a firm's formal corporate planning activities. Thus, the interest in modeling has lagged the interest in formal planning which experienced a rapid growth in the late 1950's and early 1960's [9, p. 30].

Corporate modeling has also been found to be an expensive endeavor. The construction of most models has required the expenditure of several man-years of effort. One large, sophisticated model required a total of 23 man-years for development [10, p. 44]. The large investment necessitated by a corporate modeling effort has led to the development of generalized models and to the development of specialized modeling languages.

The generalized models are usually relatively simple models which produce pro forma statements according to common accounting definitions. They allow a varying degree of freedom in the input of data but allow little or no freedom in specifying the logic of the model or the form of the output. Such models have a limited utility since, in order to gain generality, the models sacrifice detail.

Several corporations such as IBM [14, 19] , Dow Chemicals [17], On-Line Decisions [2], and Monsanto [4] have developed specialized modeling languages or modeling systems which are designed to decrease the cost of developing a corporate model. Modeling systems are not corporate models themselves, but rather assist in the development of corporate models by providing data structuring routines, predefined mathematical routines, and report writing routines. IBM's Planning Systems Generator (PSG), for example, provides a data structure which can contain over 10,000 values, provides routines for computing the depreciation and the retirement of investments, and provides capabilities for printing reports complete with titles and column headings.

These specialized languages, while representing a major improvement in modeling capabilities, have tended to be oriented more towards the computer specialist than towards the manager. Most managers would have difficulty in interpreting a corporate model when expressed in one of the specialized languages. For example, in order to use PSG the user must be capable of programming in the FORTRAN language. Thus, language development has not yet attained the desirable quality of allowing "the modeler and planner to conceptualize the simulation model in the language it is to be programmed" [15, p. 172].

Purpose of Study

The purpose of this study is to further the development of modeling languages that can be used in constructing corporate models. A modeling system consisting of three languages will be developed which will provide facilities for data structuring, logic formulation, and report writing.

The system will be comprehensible to managers so that a model will not have to be interpreted for the manager by a computer specialist. This user orientation will make the system particularly well suited for the initial modeling effort of a firm, for the development of exploratory and/or short-life models, and as a pedagogical aid in introducing managers to the concept of modeling.

Methodology

This study will review the literature to determine the characteristics of currently available modeling systems and to determine the desirable characteristics of such systems. A new modeling system will then be developed which supplies the facilities necessary to support a corporate modeling effort while retaining a high level of management orientation.

Overview

The results of this study are reported in four sections. The first chapter introduces the concept and describes the functions of corporate models. The growth of corporate modeling activity is also reviewed. The purpose of this study is given to be the development of a corporate modeling system that is suitable for use in a variety of modeling efforts including use as a pedagogical device.

The second chapter describes the desirable characteristics of a modeling system. Several modeling systems currently in use are discussed and are compared in relation to the desirable characteristics which have been defined.

The third chapter introduces the corporate modeling system which was developed in this study. General design concepts are discussed, and then the three languages which comprise the corporate modeling system are described.

The fourth chapter presents examples of the use of the modeling system discussed in the previous chapter, while the fifth chapter presents the conclusions of this study and recommendations for future research.

CHAPTER II

MODELING SYSTEMS

The initial modeling effort of most firms has been expended in the development of individualized models [see 18; 4; and 3]. The individualized models could be executed with different data, but the logic and the reports printed by the models were "set in concrete.". The logical structure and the report formats could only be altered with considerable difficulty by reprogramming the model. This was found to be an important deficiency, since model building is an evolutionary process. Models are continually altered to reflect new requirements and changing situations. As McKenney states, "An adaptable and changing model is essential if the model is to be used over an extended period of time" [15, p. 170]. Thus, the initial modeling efforts highlighted the deficiency of the individualized model approach and, in so doing, helped define the capabilities required of corporate modeling systems.

Desirable Characteristics of Modeling Systems

The previous discussion of the functions of corporate models indicated the necessity for the integration of the modeling effort with the management process. That is, the models should reflect the manager's view of the organization; and,hence, the managers should be intimately involved in the construction and operation of the models. The major difficulty encountered in accomplishing this integration has been the problem of interfacing managers and computers. McKenney summarizes the problem thusly,

> An impediment to more adequate rapport between modeler and planner is the state of present computer languages. The language of the program for the model has to be interpreted to the planner. This interpretation creates ambiguities and misunderstandings which limit the effectiveness of present simulations as a tool for most planners. Hopefully new computer languages will allow the modeler and planner to conceptualize the simulation model in the language it is to be programmed [15, p. 173].

Modeling systems consist of computer languages and data-handling routines that simplify the construction of models and facilitates the manager-computer interface. The desirable characteristics of such systems may be summarized as follows:

1. The modeling system should be comprehensible to managers.
2. The logical structure of a model should be easily translated into computer executable form.
3. The modeling system should have convenient report generation capabilities.
4. The specification and the alteration of the values of input variables should be easily accomplished.
5. The creation of the logical structure of a model, of the report format, and the value of variables should be independent.
6. The modeling system should support the performance of simple logical and arithmetical tasks as well

as providing the capability of performing special purpose calculations such as the discounting of cash flows.

7. The modeling system should facilitate the performance of sensitivity analyses.
8. The modeling system should facilitate the performance of Monte Carlo simulations.
9. The communication of the values of variables among models should be supported.
10. The modeling system itself should be alterable and expandable.

The first requirement of a modeling system is that the conceptual design of the system and the use of the system be comprehensible to managers. A manager should be able to understand the structure of the system and to develop unsophisticated models with a minimum amount of training. Also, a manager should be able to comprehend the logical structure of a model created by others. This requirement does not mean that all managers will actually be physically creating computer-based models; nor does it mean that technically competent personnel will no longer be needed. What is implied is that managers will be better able to interface with technical personnel and with computers because the modeling system will function as a common language.

In order for a modeling system to fulfill its role as a common language, it must provide convenient methods for the creation and manipulation of the three components of a model - the logical structure, the data structure, and the reports. The logical structure should be easily expressed in an "English-like" language that is both understandable by humans and executable by computers. For example, the sentence

MARGINAL_INCOME EQUALS SALES MINUS VARIABLE_EXPENSES

is comprehensible to managers and is an executable statement under the corporate modeling system introduced in the next chapter.

The data structuring capabilities are an important part of a modeling system since most models are executed over time periods rather than for just one time period. For example, a budgeting model might be executed for twelve time periods (i.e., twelve future months), while a long-range planning model might be executed over five periods corresponding to five future years. A modeling system should thus accommodate a varying sized data structure.

The need for convenient report generation capabilities might at first appear to be simply a convenience item, almost a "frill," since any computer output could be copied over into a meaningful form for managers. But a report generator is more than just a convenience item. The concept of a modeling system is to increase the integration of models into the management process by improving the interface between managers and computers. This interface is a two-way interface; the manager supplies information for the computer and gets information out of the computer. The objective is to lessen the need for intermediaries in both cases. Thus, the output should not have to be interpreted for the manager. Also, a report generator makes it easy to alter the formats of reports to reflect changes in the logical structure of models and new needs for information.

The fifth characteristic summarized above was that the three components of a model (logic, data, and report) should be created independently. This does not mean that the three components are unrelated. Rather, the three components should be created separately and then linked together when the model is to be executed. This will allow different data to be executed with the same logic, different logic to be executed with the same data, and different reports to be printed with appropriate logic-data combinations.

The sixth characteristic presented above is, of course, the primary _raison d'être_ of models and modeling systems. The purpose of a model is to perform some prescribed calculations which will assist the manager in his planning and decision-making activities. A modeling system must thus provide the capability of performing logical and arithmetical tasks and provide certain special purpose functions that are commonly required in the business environment.

The modeling system should also provide capabilities beyond simple arithmetical tasks by supplying facilities for the performance of sensitivity analyses and Monte Carlo simulations.

Sensitivity analysis requires that a model be repeatedly executed with a different value for a particular variable being used in each execution. The modeling system must therefore provide the facility for denoting the variable to be used in the sensitivity analysis as well as the successive values it is to assume.

The requirement for the support of Monte Carlo simulations is much more involved than the requirements for sensitivity analysis. Monte Carlo simulation require repeated executions as does sensitivity analysis, but it also requires two further capabilities. First, some of the variables in the model must be expressed as random variables which either conform to a mathematical distribution (e.g., a normal distribution) or conform to some empirical distribution. Secondly, the results of the repeated executions must be accumulated so that the relative frequency of the occurence of the different values for a particular variable may be determined. The report generating routines should then have the capability of depicting these relative frequencies.

A corporate model is typically composed of an interrelated set of models. For example, a corporate model may consist of a number of product models whose outputs are summarized by divisional models whose outputs are in turn summarized by a model for the total corporation. This modularization of a corporate model is desirable for a number of reasons. Modularization allows for some areas of the organization to be modeled in more detail than others, allows more decentralization in the development and use of the models, aids in the alteration and maintenance of the models, and allows the independent execution of parts of the corporate model.

The subdivision of a corporate model also gives rise to a requirement for the ability to communicate the values of variables among models. The values computed in one model must be saved in such a manner that they may be accessed by a model which is executed later. In this way models for different products may be executed with their results being saved and later accessed by models which summarize the performance of each division. Likewise, a model could access values computed by the divisional models and then summarize the performance of the total organization.

A final requirement of a modeling system is that it should itself be amenable to alteration and expansion. As a modeling system is used and as the modeling effort becomes more sophisticated, it is likely that there will be a need for capabilities which were not foreseen at the time of the original development. This growth has been exemplified by IBM's PSG which was revised to increase the number of special purpose mathematical routines and to improve the data strucuturing capabilities. Likewise, the Dow Chemical Company's PS.1 has a history of continual alteration and expansion and is currently being completely rewritten [17]. The original development of a modeling system is the beginning rather than the end. Thus, a modeling system should be designed for alteration and expansion.

State of the Art

Although a number of corporate models and modeling systems have been reported in the literature, little detailed information has been reported. Gershefski has presented in detail the Sun Oil Company's corporate model [10]. Then later, in studying the state of the art, he concentrated more on general attributes such as the number of firms engaged in the modeling activity, the organization and resources required for a successful modeling effort, and the general characteristics of models [9]. Dickson, Mauriel, and Anderson have summarized with a bit more detail their study of twenty models [5]. But, in general, corporate modeling activity has not been reported in much detail.

In addition to the lack of reported detail, another difficulty in determining the state of the art is the lack of distinction in the literature between <u>models</u> and <u>modeling systems</u>. Dickson <u>et</u> <u>al</u>. distinguish between two "philosophies" of models - the rigid structure approach versus the flexible structure approach - but fail to clearly distinguish modeling systems.

> One philosophy exemplified by our findings noted above is to adopt a fixed structure model approach which forces the user to employ existing accounts, fixed output reports, and a limited set of options for attaching values to variables in the model. There is an alternative philosophy--namely, to build a more general and flexible model which allows the user considerable latitude in choosing model variables and methods for setting their values over the projected horizon [5, p. 53].

Rigid or fixed structure models predefine the variables which are used in the model, predefine the method for assigning values to the variables, and predefine report formats for the printing of variables. Most corporate models appear to be of this type [5, p. 58].

A rigid structure model is difficult to alter in response to changing situations so that it is desirable to have a more flexible model structure. Flexibility is usually obtained by allowing the user to specify the "names" of the variables used in the model, by allowing the user to design the format of reports, and by allowing alternative methods for assigning values to variables (e.g., the explicit statement of all values, the specification of beginning values and growth rates, or the specification of a variable as a function of another variable). Flexibility is also obtained by allowing the input of values to override values which would normally be calculated. Using these techniques, considerable flexibility can be built into a model in that the meaning of the variables, the logic of the model, and the reports produced by the model can be altered.

There are a number of models of both the rigid and flexible structure type reported in the literature. Two easily compared models which are described in sufficient detail in the literature are the rigid structure model developed by Dinter [6] and the flexible structure model FINAN$ developed by General Electric [8]. Both models are interactive financial models which produce *pro forma* income statements, balance sheets, ratio analysis, etc. The logic in both models is based upon general accounting definitions.

In the Dinter model the logic, variables, and reports are predefined. The user interacts with the model by specifying the values of variables and then noting the response evidenced by the printed statements. The user can then change some of the values of the variables and re-execute the model. The user is assisted in this process by exception reports which highlight "out of bounds" conditions and list possible causes for the exceptions.

The General Electric model does not have the exception reporting feature but does allow the user more flexibility in the application of the model. For example, the values of variables may be assigned in several different ways. There are three options for specifying the values of anticipated sales and four options for specifying the values of other accounts. Two of these four options allow the user to express the value of an account as a function of sales and/or the change in sales. Thus the user has a limited ability to create some of the logic of the model. There is also flexibility, though limited, in the specification of reports. The typesof reports and their formats are predefined, but the user can specify the title that is printed for each account.

A modeling system is different in philosophy from either a rigid or flexible structured model. A modeling system is a set of special purpose computer languages and facilities which facilitate the construction, alteration, and execution of models. It is not itself a corporate model; it is used in the construction of corporate models.

A number of corporate modeling systems have been developed and reported in the literature. Several of these systems are reviewed below with the intent of providing a representative though not exhaustive view of the available corporate modeling systems.

Perhaps the most widely used corporate modeling system is the Planning Systems Generator (PSG) developed by Henry Lande of IBM [14]. PSG consists basically of data input and data structuring routines, report generation routines, and arithmetic functions which can be incorporated into a corporate model. The logic of a model is expressed in the FORTRAN computer language. PSG is, undoubtedly, one of the best corporate modeling systems currently available. It does, however, have some limitations.

The foremost shortcoming is that the modeler must be capable of programming in the FORTRAN computer language. FORTRAN is not an extremely difficult language to learn but it is algebraically oriented rather than being "English" oriented. This is particularly bothersome in that variables must be referenced by position rather than by name. For example, the value of sales cannot be referenced by simply refering to "SALES," rather the position in a table of values must be known so that the values can be referenced by specifying the appropriate line number in the table. The user must thus coordinate the values, the positions, and the meanings of variables.

Two other limitations of PSG are that there are no built-in capabilities for the performance of sensitivity analyses and Monte Carlo simulations. Sensitivity analysis can, of course, be accomplished by the user repeatedly initiating the execution of a model after he has altered the value of a variable. It is more desirable, however, for the user to be able to initiate execution only once and have the system automatically perform the sensitivity analysis for a designated variable.

The support required for Monte Carlo simulation is more involved and cannot easily be compensated for by simple repetitive executions. Monte Carlo simulation requires many more repetitions than does a sensitivity analysis. It is therefore desirable for the modeling system to be capable of performing Monte Carlo simulations. This requires the capability of expressing variables as random variables, the capability of selecting a value of the random variables to be used in the calculations, and the capability of counting the number of occurences of various outcomes. These capabilities are not supported by PSG.

Another modeling system, the Financial Analysis and Planning System (FAPS), developed by On-Line Decisions, Inc. has capabilities similar to PSG [2]. In addition, FAPS has the capability of performing sensitivity analyses and has extensive capabilities for the statistical and mathematical manipulation of time series data. But, just as with PSG, the user must reference variables by position rather than by the use of a meaningful name; and FAPS does not support Monte Carlo simulations.

The Dow Chemical Company has developed a modeling system called Planning Simulator 1 (PS1) in which variables may be referenced either by position or by a one to six character name [17]. PS1 is not, however, as versatile a system as PSG or FAPS.

One of the major inconveniences of PS1 is that arithmetical operations cannot be performed in series. For example, the PS1 statement

Y = ADD(A,B)

denotes that the variable Y should assume a value equal to the value of A added to the value of B. Likewise, the statement

Z = SUB(Y,W)

denotes that the variable Z should assume a value equal to the value of Y minus the value of W. Thus Z would have the value of A plus B minus W. But the operations "ADD" and "SUB" cannot be used in series to compute the value of Z directly. Consequently, either all computations would have to be broken down into single operations or a multitude of special operations must be created. PS1 has evolved in accordance with the latter approach so that there are currently over one hundred operations in the PS1 language. To become proficient in the use of the language, the user would have to familiarize himself with each of these operations.

Another problem in using PS1 is that there appears to be no convenient method for saving the results from the execution of a model and then later accessing the results as inputs into another model. Values can be transferred from one model to another only when the executions of the models are contiguous. This is a significant handicap in corporate modeling since a set of models is often used to summarize plans on a product-division-corporation continuum. Without the capability of permanently saving results, the entire set of models must be run whenever any change is made.

Another shortcoming of PS1 is that the specification of report formats is integrated with the specification of logic. Thus, the user is not free to mix logic, data, and report modules at the time of execution. And, if a report is to be changed, the logic-report module must be altered.

Finally, PS1, like IBM's PSG, lacks the capability to automatically perform sensitivity analyses and Monte Carlo simulations.

A fourth modeling system, the CAPEX Corporation's AUTOTAB is an interactive system available on General Electric's Time Sharing Service [1]. AUTOTAB is a relatively simple system to use and has the advantages of referencing variables by name and transferring values among models by storing and retrieving them from permanent files.

One limitation of AUTOTAB is that reports, data, and logic are contained in a single "package" so that the "package" must be altered to change any of the components. Also, a common report could not be used with two different logic components without duplicating the report specifications.

In addition, AUTOTAB does not support sensitivity analyses and Monte Carlo simulations.

A final modeling system, reported by Buchman, is Monsanto's APEX system [4]. APEX was designed for use in creating special purpose models rather than for use in building a total corporate model. The system seems to be an excellent system for its intended use. It obtains a higher level of user orientation than all the other systems mentioned except AUTOTAB. It is simple to understand and easy to use and yet offers advanced capabilities for sensitivity analyses and Monte Carlo simulation.

APEX is not as well suited, however, for use in corporate modeling when an interrelated set of models is used. APEX does not allow the independent creation of logic, data, and report modules and does not support the transfer of values among models.

The previous discussion of five representative modeling systems is summarized in Table 1. The systems are compared in terms of a few major design concepts with no attempt being made to discuss the details of implementing any of the systems. However, the reported usage of PSG and FAPS has shown that sophisticated modeling systems can be incorporated into planning and decision-making activities and can make significant contributions to these efforts. On the other hand, systems such as AUTOTAB and APEX have demonstrated the desirability and practicality of maintaining a high level of user orientation. The next chapter describes a modeling system that attempts to combine these two lessons.

TABLE 1

Summary of Characteristics of Five Modeling Systems

	Systems				
Characteristics	PS1	PSG	AUTOTAB	FAPS	APEX
Variables Referenced by Descriptive Names	Yes	No	Yes	No	Yes
Reports and/or Data Created Independently of Logic	No	Yes	No	Yes	No
Built-in Capability for Sensitivity Analysis	No	No	No	Yes	Yes
Built-in Capability for Monte Carlo Simulations	No	No	No	No	Yes
Convenient Communication of Values Among Models	No	Yes	Yes	Yes	No
Convenient Specification of Arithmetical Tasks	No	Yes*	Yes	Yes**	Yes

*User must be capable of programming in the FORTRAN language.

**FAPS uses a BASIC-like language.

CHAPTER III

A CORPORATE MODELING SYSTEM

This chapter describes a corporate modeling system named CMS/1. The structure and the operating characteristics are described first. Second, the major design concepts of the system are discussed. Finally, more detailed explanations of the characteristics and the use of the system will be given.

This chapter is intended to be an introduction to the concepts, capabilities, and use of CMS/1. It is not intended to supply all of the details necessary for the use of CMS/1. A USER'S MANUAL which does supply such detailed information is contained in Appendix A.

Overview of CMS/1

The logical organization of CMS/1 is depicted in Figure 1. The user originates logic, data, and report modules in the appropriate CMS/1 languages, i.e., the logic and data specification language or the report format specification language (see Figures 2, 3, and 4). Then the modules are entered, usually via cards, into CMS/1 where they are translated into a more conveniently executed form and stored on a magnetic disk. The user can then give a command for specific logic, data, and report modules to be retrieved from the disk, combined into a complete model, and executed.

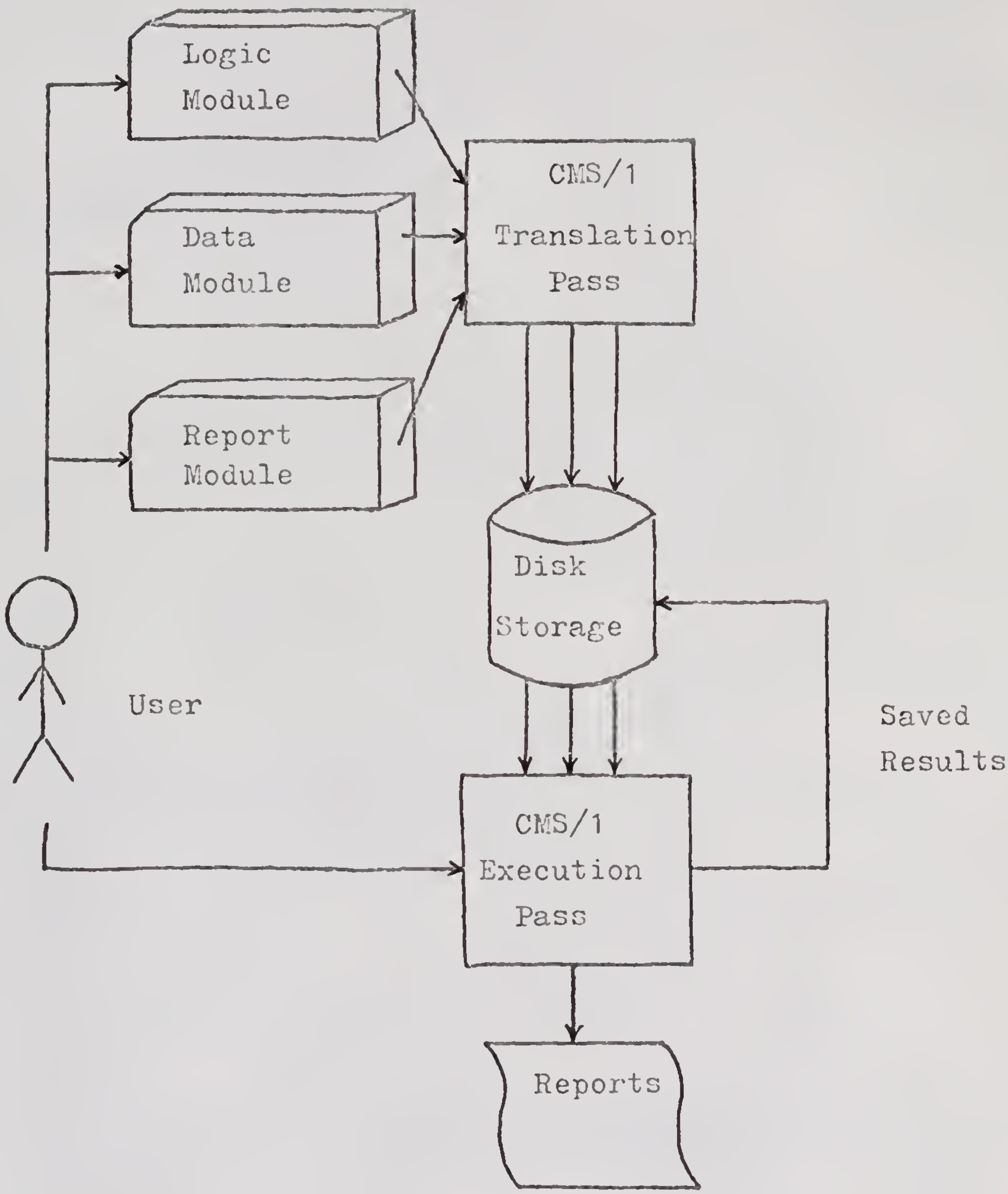


Figure 1. Logical organization of CMS/1.

```
*DATA PRODUCT_DATA  PERIODS 1 TO 5
 SET INVESTMENT EQUAL TO 9000000, 0, 0, 0, 0
 SET SALVAGE EQUAL TO 0, 0, 0, 0, 4250000
 SET TAX_RATE EQUAL TO .48
 SET PRICE EQUAL TO 516
 SET FIXED_COST EQUAL TO 306350
 SET VARIABLE_COST_RATE EQUAL TO 421.65
 SET BEGINNING_MARKET_SIZE EQUAL TO 220000
 SET GROWTH_RATE EQUAL TO  , .08, .06, .03, .02
 SET BEGINNING_SHARE EQUAL TO .01
 SET MAXIMUM_SHARE EQUAL TO .15
 SET DISCOUNT_RATE EQUAL TO .12
 SET DEPRECIATION EQUAL TO DEPRECIATE(1,INVESTMENT,SALVAGE)
```

Figure 2. Sample data module.

```
*LOGIC
 SET MARKET_SIZE EQUAL TO GROWTH(BEGINNING_MARKET_SIZE, ,
1                     GROWTH_RATE)
 SET MARKET_SHARE EQUAL TO LINEAR(BEGINNING_SHARE,
1                     MAXIMUM_SHARE,)
 SET UNITS_SOLD EQUAL TO MARKET_SIZE TIMES MARKET_SHARE
 SET SALES EQUAL TO UNITS_SOLD TIMES PRICE
 SET VARIABLE_COSTS EQUAL TO VARIABLE_COST_RATE TIMES
1                     UNITS_SOLD
 SET TOTAL_COSTS EQUAL TO FIXED_COST PLUS VARIABLE_COSTS
1          PLUS DEPRECIATION
 SET OPERATING_INCOME EQUAL TO SALES MINUS TOTAL_COSTS
 SET TAX_EFFECT EQUAL TO TAX_RATE TIMES DEPRECIATION
 SET CASH_FLOW EQUAL TO OPERATING_INCOME PLUS SALVAGE
1          PLUS DEPRECIATION PLUS TAX_EFFECT
 SET PRESENT_VALUE EQUAL TO DISCOUNT(CASH_FLOW,
1                       DISCOUNT_RATE)
 SET PROFIT_INDEX EQUAL TO PRESENT_VALUE DIVIDED BY
1                     INVESTMENT
 SET ROI EQUAL TO INTRL_RATE(INVESTMENT, CASH_FLOW)
```

Figure 3. Sample logic module.

```
*REPORT  NEW_PRODUCT
 TITLE                         ANALYSIS OF NEW PRODUCT
 MARGIN  0
 COLUMN SIZES 15, 0, (9)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT",  "   1972",  "   1973",
1                 "   1974",  "   1975",  "   1976"
 COLUMN HEADINGS "--------", ("   ----")
 ITEM "INITIAL MARKET", BEGINNING_MARKET_SIZE
 ITEM 2, GROWTH_RATE
 ITEM 2, BEGINNING_SHARE
 ITEM 2, MAXIMUM_SHARE
 SKIP 1 LINE
 ITEM INVESTMENT
 ITEM DEPRECIATION
 ITEM "SALVAGE VALUE", SALVAGE
 SKIP 1 LINE
 ITEM MARKET_SIZE
 ITEM "SHARE OF MARKET", 3, MARKET_SHARE
 ITEM UNITS_SOLD
 SKIP 1 LINE
 ITEM "PRICE PER UNIT", 2, PRICE
 ITEM SALES
 ITEM VARIABLE_COSTS
 ITEM FIXED_COST
 ITEM TOTAL_COSTS
 ITEM  "NET INCOME", OPERATING_INCOME
 SKIP 1 LINE
 ITEM CASH_FLOW
 ITEM PRESENT_VALUE
 ITEM 2, PROFIT_INDEX
 ITEM 3, ROI
```

Figure 4. Sample report module.

The execution of a model is actually a six-step process.

1. The first step is to examine the specified logic module and report modules to determine the names of all variables which will be used.
2. The specified data modules are retrieved and the values are assigned to the indicated variables. Variables are ignored if they are contained in a data module but are not contained in the logic module or in a report module.
3. The third step is to perform the calculations specified in the logic module.
4. If a sensitivity analysis or Monte Carlo simulation is in progress, the second and third steps are repeated.
5. If requested, the values of specified variables are saved by creating a new data module.
6. The requested reports are printed (see Figure 5).

General Design Concepts

The design of CMS/1 is based upon two basic assumptions. They are:

1) It is both necessary and desirable to improve the interface between managers and computers with respect to the creation and manipulation of corporate models.
2) Corporate models are basically data manipulators. Data must be acquired, "massaged," stored, and printed out in reports.

Starting with these two assumptions about corporate models and their environments, a number of more specific

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2200	10692	20148	29832	39690
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5517067	10396593	15393451	20479968
VARIABLE COSTS	927630	4508277	8495586	12578773	16735228
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	5764627	9751936	13835123	17991568
NET INCOME	-1048780	-247560	644657	1558328	2488400
CASH FLOW	357220	1158439	2050656	2964327	8144399
PRESENT VALUE	9207309	*	*	*	*
PROFIT INDEX	1.02	*	*	*	*
ROI	0.172	*	*	*	*

Figure 5. Output from sample model.

design characteristics can be derived. The design characteristics may be grouped into four catagories: 1) the "point of view" represented in the design of the system, 2) characteristics which promote ease of use, 3) characteristics which provide flexibility, and 4) technical characteristics.

The primary point of view reflected in the design of a modeling system should be that of the manager and not the computer specialist's point of view. This seems self-evident, but it is a difficult distinction to maintain. It is tempting to design the modeling system from the standpoint of the ease with which the system itself can be constructed rather than from the standpoint of the ease with which the system can be utilized. The designer must ask himself "What facilities are needed by the modeler?" and not just "What facilities can be easily provided?" In order to be better able to maintain this distinction, CMS/1 is written in the computer language PL/1. PL/1 is a very powerful computer language developed by IBM. It has advanced capabilities that are not available in other languages such as FORTRAN and BASIC which have been used most often in the development of modeling systems. The use of PL/1 greatly reduces the number of restrictions imposed on the modeling system by computer programming considerations.

A second characteristic of the modeling system is that the system should be easy to use. One of the major contributants to the ease of use is the utilization of "English-like" languages for communication between the manager and the computer. The objective is to design a satisfactory language which falls between the extremes of the verbose, sometimes imprecise English language and a cryptic, though precise, numerical code. But the perceived successfulness of this endeavor depends in part on the user. An experienced user usually prefers a less verbose language than does a novice. For this reason, CMS/1 is designed with both "English-like" forms and short-forms that can be used to designate a required operation.

Another contributant to ease of use is the existence of default conditions. A system that is designed to be used in corporate modeling must be capable of performing many different tasks. However, flexibility increases the difficulty of use since the user must denote which alternative he wishes to select for each option that is available to him. This problem may be mitigated by the use of defaults. If no alternative is specified for an option, a predefined alternative is assumed by the system. The employment of selection by default greatly simplifies the use of a system.

Another contributant to ease of use is the provision of extensive error checking routines. When an error is detected, it should be diagnosed and noted by the printing of a precise and clearly worded error message. Good diagnostic capabilities are an important part of a modeling system. The system should be designed to assist the user in identifying his mistakes as well as being designed to execute error-free input.

A third design characteristic is flexibility. Modeling systems should be flexible enough to be used by different organizations and to be used for different purposes. They can be used in preparing budgets, in cash flow analyses, in facilities planning, in long-range planning, and in special projects such as capital investment analysis.

One method of providing the needed flexibility is to provide the capability of creating models in segments. A model is segmented into modules which contain only data, modules which contain only report formats, and a module which specifies the logical relationship among the variables. Then, when the user wishes to execute a model, he can select the appropriate modules and have them combined and executed.

There are several significant effects of this modular approach. First, the logic module can be created totally independently of the number of periods over which it will be executed. A planning model could thus be executed over three, five, ten, or any other number of years without changing the logic of the model. The flexibility in the specification of time periods also makes possible the development of models which have different time horizons. For example, budgeting is usually concerned with twelve monthly periods, while long-range planning is typically concerned with a five-year period, and capital investment decisions may cover widely different periods. A modeling system which supports varying horizons can be used in each of these areas.

Second, the logic module can be created without regard as to whether it will be executed as a deterministic or as a stochastic model, or whether it will be executed in conjunction with a sensitivity analysis. These alternatives are a function of the data which are used and the manner in which the model is executed; they need not be reflected in the logic of the model. For example, the statement

DOLLAR_SALES EQUAL UNIT_SALES TIMES UNIT_PRICE

defines the logical relationship between three variables. If the two variables "UNIT_SALES" and "UNIT_PRICE" both are single-valued variables, then the relationship is deterministic. But, if the variable "UNIT_SALES" is defined to be a random variable, then the relationship becomes stochastic. However, the stated relationship has not been changed; only the values assigned to the variables have been altered.

A third implication of the modular construction of models is that multiple report and data modules can be combined with one logic module. Multiple report modules can be used to print out different reports or to tailor the same basic report for different users. Multiple data modules can be used to incorporate into a model the results saved previously by other models. The use of multiple data modules also means that all the period values for variables need not be contained in a single data module. That is, budgeted values for the current year might be stored in one data module with estimated values for the next five years stored in another data module. The data modules could then be combined with logic and report modules to produce reports covering the current year and a five-year plan.

A final implication of the modular approach is that the modeling system must maintain the modules on computer accessible storage devices. This is required since the modules are created independently and a combination is selected for execution at a later time. The user can therefore readily access models kept on magnetic disks instead of primarily using cards as a storage medium.

Another method for increasing flexibility is the referencing of variables by user-supplied names. The use of names complements the use of modules. A modeler simply uses the variable's name in different modules. When the modules are combined for execution, the like names can all be linked by the modeling system to the same values. When positions instead of names are used to identify variables (as in PSG), the user must insure the positional equivalence of variables among modules and/or models that are to be interrelated.

Finally, the flexibility of a modeling system can be increased by allowing arithmetic operations to be used in the specification of data modules and by allowing values specified in a data module to supersede values calculated in a logic module. The use of arithmetic operations in defining data means that the statement

COST_OF_GOODS_SOLD EQUALS SALES TIMES .75

could be used in a data module to define the values associated with the account "COST_OF_GOODS_SOLD." Of course, the values associated with "SALES" would have to be previously defined. The second method, the superseding of values calculated in the logic module, means simply that if a value for a variable is given in a data module, then any equation appearing in the logic module that would alter the value is ignored. These two methods provide the user great flexibility in temporarily altering the logical structure of a model without physically altering the logic module. The user can accomplish the alterations by simply temporarily adding to the model a data module consisting only of the desired changes.

The last classification of design characteristics is composed of two technical considerations. The first is the consideration of size limitations placed on the model, and the second deals with the procedure used in executing a model over several time periods.

CMS/1 is designed so that a user will be restricted by the physical capacity of his computer rather than by the modeling system. Storage areas in CMS/1 are allocated dynamically so that capacity is limited only by the "counters" which are used to reference the storage areas. Consequently, a model may contain as many as 32,767 different variables. Also, each variable may be defined over a maximum of 32,767 time periods. Thus, CMS/1 has a capacity of over one billion values. This compares to IBM's PSG which has a capacity of less than eleven thousand values. The number of statements which may be included in a model is a function of the size of the statements. However, over five million "average-sized" statements could easily be accommodated. All of these limitations are much in excess of the physical capacity of current computing equipment.

The second technical consideration deals with the procedure used in executing a model over several time periods. A model could be executed from beginning to end for the first time period, executed again for the second time period, and so on until all time periods have been accounted for. Or, each equation could be executed in turn over all time periods. That is, the first equation would be executed over each time period, then the second equation, and so on until all equations have been executed. A third alternative would be to execute in turn each operaion within an equation over all time periods. Thus, if the statement

SET INTEREST_EXPENSE EQUAL TO LOANS TIMES INTEREST_RATE

were executed over twelve time periods, the multiplication would be carried out twelve times; and then the twelve products would be assigned to the variable "INTEREST_EXPENSE."

The third alternative would actually yield the most rapid execution since CMS/1 is executed by an interpreter rather than by being compiled. This procedure has, however, some undesirable characteristics. The procedure cannot be used 1) when a variable is a function of its own values in previous time periods, 2) when a variable is a function of another lagged variable and the equation defining the first variable precedes the equation defining the variable which is lagged, and 3) when a condition which exists in one time period alters the sequence in which the equations are executed.

The first problem can be avoided by using the second procedure described above, i.e., the execution of each equation in turn over all time periods. But the second and third problems would remain. All three problems can be avoided by executing the complete model over each time period. This, however, is accomplished at the expense of longer execution times. Thus, this execution procedure is followed by CMS/1 only when one or more of the three problems actually occurs. In all other cases, CMS/1 uses the more efficient procedure of executing in turn each operation within an equation over all time periods.

This section has discussed the major design concepts of CMS/1. The next three sections will discuss the threc CMS/1 languages which embody these concepts.

Control Language

The purpose of the control language is to allow the user to direct the activity of the modeling system. It is through the use of the control language that the user instigates the creation of modules and the execution of models.

The general form of a control language statement is

```
*KEYWORD  parameter-list
```

Each control statement begins with an asterisk in the first position followed by none, or one or more blanks. The first non-blank characters encountered after the asterisk must be one of seven control keywords. Each control keyword uniquely designates an activity performed by CMS/1. Following the control keyword and separated from it by blanks is a list of parameters. The parameter list provides additional information which is needed to carry out the activity which has been requested. The parameter list is composed of sets of parameter keywords and parameter values. Each set must be separated from surrounding sets by a comma or by blanks.

In the following discussion and examples of control language statements, keywords will be denoted by capital letters whereas user-dependent information will be denoted by small letters.

There are five types of activities which the user can specify. The user can specify that a module (logic, data, or report) is to be created, that modules are to be combined and executed, that modules are to be destroyed, that no action is to be taken (i.e., the null activity); and, the user can govern whether or not the logic, data, and report statements he supplies to CMS/1 are to be printed.

The first type of activity is the action of creating a module. A statement such as

```
*LOGIC
```

signifies that the following statements conform to the requirements of the logic specification language. The statements are read by CMS/1, checked for errors, and translated into a more easily executed form. The translated form is then temporarily stored on a magnetic disk. If the user wishes to permanently store the logic module, a name for the module is specified as a parameter on the LOGIC statement. The following statement specifies that a logic module named "sample" is to be created and saved.

```
*LOGIC  sample
```

The statements for creating data and report modules are analogous to the LOGIC statement. That is,

```
*DATA  budget
```

and

```
*REPORT  balance_sheet
```

would be used, respectively, in creating a data module named "budget" and a report module named "balance_sheet."

The DATA statement may also contain the parameter keyword PERIODS followed by numbers representing the first and last periods for which data is given. For example, either the statement

```
*DATA  budget, PERIODS 1970 TO 1972
```

or

```
*DATA  budget, PERIODS 1970, 1972
```

could be used to create a data module named "budget" containing values for the years 1970, 1971, and 1972. If the periods parameter is not given, CMS/1 assumes the specification

```
PERIODS 1 TO 5
```

The second type of activity the user may specify is the activity of combining and executing modules. This activity is accomplished by the use of the EXECUTE statement. The simplest form of the statement is

```
*EXECUTE
```

The above statement would cause the most recently created unnamed logic, data, and report modules to be combined and executed. If an unnamed logic module does not exist, an error message is printed. If an unnamed report module does not exist, a default report format is used and the values of all variables defined in the logic module are printed.

Named (i.e., permanently saved) logic, data, and report modules may be combined for execution by use of the parameter keywords LOGIC, DATA, and REPORT in the following manner.

```
*EXECUTE LOGIC sample, DATA budget, REPORT balance_sheet
```

Multiple data and report modules can also be specified.

```
*EXECUTE LOGIC sample, DATA budget_70, budget_71,
1        REPORT balance_sheet, income_statement
```

Multiple reports are printed in the same sequence in which their names appear in the REPORT parameter set. Multiple data modules are "overlayed" on each other in the same sequence in which their names appear in the DATA parameter set. Thus, if a variable is assigned values in several different data modules, the resulting values of the variables will be the values assigned by the data module appearing latest in the sequence specified by the DATA parameter set.

If a named logic module is specified, an unnamed logic module is, of course, not used. Likewise, the default report format is not used if a named report module is designated. However, the most recently created unnamed data module is always used. It is incorporated into a model as though it were the _last_ named data module in the DATA parameter set. Consequently, it overlays all named data modules.

Another parameter set that may appear on an EXECUTE statement specifies the number of times the model is to be executed in a sensitivity analysis or in a Monte Carlo simulation. The form of the parameter set is

ITERATIONS n

where n is the number of times the model is to be executed. The value of n must be between 1 and 32,767. If this parameter set is not given, n is assumed to have a value of 1.

An additional parameter set that may be used in conjunction with Monte Carlo simulations specifies the "seed" number for a pseudo random number generator. The form of the parameter set is

INITIAL RANDOM NUMBER IS n

or

INITIAL n

where n is the value to be used as the seed. The seed should be an odd number containing at least five digits. Its value must be between 1 and $2^{31}-1$. If a Monte Carlo simulation is performed without specifying a value for the seed, a value of 65,549 is assumed.

CMS/1 allows the saving of values computed in one logic module for later use in another logic module. The values are stored in a data module that is created by CMS/1. The user may name this data module by specifying the desired name in a SAVE parameter set. The EXECUTE statement

*EXECUTE LOGIC product SAVE results

would execute the logic module named "product" along with the most recently created unnamed data module. The default report would be printed, and the values of variables saved by the logic module "product" would be stored in a data module named "results."

If variables are saved by a logic module when the SAVE parameter set is not given on the EXECUTE statement, the resulting data module is given a name created from the logic module name and the date.

Finally, headings that are to be printed at the top of each report may be given on the EXECUTE card. Two headings may be given - the first is printed in the center at the top of each page of output, and the second is printed against the right margin at the top of every page. The form of the

HEADINGS parameter set is

    HEADINGS "first heading" "second heading"

If only one heading is specified, it is assumed to be the first heading.

The third type of activity the user may request of CMS/1 is the destroying of named modules. The following DESTROY statement would erase from the magnetic disk a logic module named "sample," a data module named "budget," and a report module named "balance_sheet."

    *DESTROY LOGIC sample, DATA budget, REPORT balance_sheet

More than one logic, or data, or report module can be erased by simply listing the module names sequentially after the appropriate parameter keyword. The statement

    *DESTROY LOGIC sample, product

would cause the logic modules named "sample" and "product" to be erased.

The fourth type of activity - the governing of the printing of logic, data, and report statements - is controlled by the PRINT statement. The user can have printed all statements received by CMS/1 or can delete the printing of logic, data, and/or report specification statements. Normally, all statements received by CMS/1 are printed. To stop the printing of logic, data, and report statements, a PRINT statement with no parameters would be used.

*PRINT

To print only data statements, the PRINT statement

*PRINT DATA

would be used. To begin printing logic, data, and report specification statements again, the statement

*PRINT LOGIC, DATA, REPORT

or

*PRINT ALL

would be used.

The final control language statement, the CONTINUE statement, initializes no system activity. However, it can terminate previous system activity. For example, the two-statement sequence

```
*DATA
*CONTINUE
```

begins the creation of an unnamed data module and then terminates the module. The two statements thereby assure that the most recently created unnamed data module is empty. This can be useful since the most recently created unnamed data module is always included in the executed model.

This section has discussed the control language of CMS/1. The keywords for each statement in the language are summarized in Table 2. The next section describes the logic and

TABLE 2

Summary of Control Language Keywords

Control Keyword	Parameter Keywords
LOGIC	
DATA	PERIODS
REPORT	
EXECUTE	LOGIC, DATA, REPORT, PERIODS, ITERATIONS, SAVE, HEADINGS, INITIAL
DESTROY	LOGIC, DATA, REPORT
PRINT	ALL, LOGIC, DATA, REPORT
CONTINUE	

data specification language.

Logic and Data Specification Language

The contents of a logic and of a data module are expressed in the logic and data specification language. The language is composed of four types of statements - assignment, group, control, and null statements. Only the first two types may be used in creating a data module; all four types may be used in creating a logic module.

The first type of statement, the assignment statement, is used to calculate the value of an arithmetic expression and then to relate this value to a named variable. The assignment statement is a very powerful and flexible instrument for performing calculations. Among the operations which can be performed are addition; subtraction; multiplication; division; and predefined procedures for determining depreciation, present values, and rates of return.

An example of the simplest form of an assignment statement is

```
SET SALES EQUAL TO 500
```

In this simple case no arithmetic is performed; the value 500 is just related to the name SALES. Then, if the next statement executed were

```
SET MISC_EXPENSE EQUAL TO SALES TIMES .05
```

the variable MISC_EXPENSE would be assigned a value of 25.

Since CMS/1 executes a model over multiple periods, the assignment statement may specify multiple values for a variable. The statement

```
SET SALES EQUAL TO 500, 600, 650, 700, 750
```

defines a value of SALES over a five-period horizon. If a model containing this statement were executed over a ten-period horizon, the values for the last five periods would be undefined. This type of statement may be used to specify the initial values of variables in data modules with the succeeding values being calculated by a logic module.

If a variable is to have the same value in all periods, then the value need be expressed only once. Hence, the following two statements are equivalent over a five-year horizon.

```
SET SALES EQUAL TO 600, 600, 600, 600, 600
SET SALES EQUAL TO 600
```

Note, however, that the second statement assigns a value of 600 to SALES over any horizon.

A second type of data and logic specification statement is the GROUP statement. This statement is used to associate a number of variables with a common group name. The whole group of variables can then be referenced by simply using the group name. For example, assuming that the variables LABOR, MATERIALS, OVERHEAD, and SALES have all been assigned values, the second statement of the two following statements would sum the values of LABOR, MATERIALS, and OVERHEAD; subtract that sum from the values of SALES; and associate the resulting value with OPERATING_INCOME.

```
GROUP  LABOR, MATERIALS, OVERHEAD UNDER MFG_EXPENSES
SET  OPERATING_INCOME EQUAL TO SALES MINUS MFG_EXPENSES
```

The use of a group name ~~i~~elicits a different response when it represents variables whose values are being computed. This use of a group name is exemplified by the following three statements.

```
 GROUP LABOR, MATERIALS, OVERHEAD UNDER MFG_EXPENSES,
1      STANDARDS
 SET MFG_EXPENSES EQUAL TO SALES TIMES STANDARDS
 SET OPERATING_INCOME EQUAL TO SALES MINUS MFG_EXPENSES
```

The second of the three above statements is executed by multiplying the labor standard times the value of sales, giving the value of the labor component of the manufacturing expense; then the materials standard is multiplied times the value of sales, giving the value of the materials used in manufacturing; finally, the standard overhead rate is multiplied times the value of sales, giving the overhead absorbed by the units sold. The procedure for executing the third statement is the same as in the previous example.

A single element (i.e., variable) of a group can be referenced by "qualifying" the element name with the group name. The previous group statement defines two groups with three elements each giving a total of six variables. References to the individual variables is accomplished as follows:

```
MFG_EXPENSES:  LABOR
MFG_EXPENSES:  MATERIALS
MFG_EXPENSES:  OVERHEAD
STANDARDS:  LABOR
STANDARDS:  MATERIALS
STANDARDS:  OVERHEAD
```

In this manner the variables may be used individually as well as by groups.

The third type of statement, the control statement, may be used only in a logic module. There are three statements of the control type - two which control the sequence in which statements are executed, and a third which terminates the execution of a logic module.

In CMS/1 as in most computer languages, statements are normally executed in the sequence they are encountered. However, the sequential execution can be altered by "jumping" over statements. The execution can be jumped forward down the list of statements or backwards to a previous statement. The method of exercising this type of control is illustrated by the following sequence of statements.

```
:
JUMP TO COMPUTE
SET RATIO)  SET RATIO EQUAL TO .25
COMPUTE)  SET ACCOUNT EQUAL TO AMOUNT TIMES RATIO
:
```

The first statement in the sequence causes the second statement to be bypassed with control being transferred to the statement named "COMPUTE." The second statement could be reached via a JUMP TO statement appearing at another point in the logic module. However, if the second statement did not have a name, it could never be reached so an error message would be printed at the time the logic module is created.

Sequential execution may also be altered by the conditional statement. The conditional statement performs a comparison and then, depending on the results of the comparison, executes one of two statements. For example, the following two conditional statements can be used as a simple means of estimating corporate income taxes.

```
IF NET_INCOME IS GREATER THAN 25000 THEN SET TAXES EQUAL
1      TO NET_INCOME TIMES .48 MINUS 6500 ELSE SET TAXES
1      EQUAL TO NET_INCOME TIMES .22
IF NET_INCOME IS LESS THAN 0 THEN SET TAXES EQUAL TO 0
```

The second conditional statement does not specify an alternative if the net income is not less than zero. In this event, the value of TAXES computed in the first conditional statement would not be altered.

The third type of control statement terminates the execution of the logic module. The form of the statement is simply

```
STOP
```

A logic module does not have to contain a STOP statement. Execution will terminate when the end of the sequence of statements is reached.

The final type of statement in the logic language is the null statement. The null statement consists solely of a statement name. For example,

START)

and

FINISH)

are valid null statements.

The null statement is used to designate a point in the logic module to which control may be passed by a JUMP TO statement.

Sample logic and data modules which demonstrate the use of some of the previously discussed statements are presented in Figures 2 and 3, respectively.

Report Format Specification Language

The third language in the corporate modeling system is used to describe the desired form of reports. The language's statements allow the user to specify the size of the printed page; the number, size, and headings for the columns of values; the spacing between lines of information; and the information that is to be printed on each line of the report. In addition, the report headings, if any, that were given in the EXECUTE control statement are printed at the top of each page.

The page size is determined by three statements. The default form for each of the statements is given below. The

PAGE LENGTH statement defines the maximum number of lines that are to be printed on a page excluding the lines contained in a footing, if one is utilized. The LINE LENGTH statement defines the maximum number of characters, including blanks, that may be contained in a printed line. A printed line may begin in the first physical print position, or it may begin to the right of the first position. The extent of indentation is governed by the MARGIN statement. The margin may be changed within a printed page. Normally, the page and line length should remain constant for a given page.

```
PAGE LENGTH  60
LINE LENGTH  130
MARGIN  2
```

The number of columns contained in a report is equal to the number of periods over which the model was executed, plus two. The two additional columns contain a description of the variable whose values are being printed and a description of the units of the values (e.g., tons, gallons, or dollars).

The width and the headings of columns are specified in statements of the types shown below. If the sum of the columns' widths is greater than the line length specified, the report is automatically continued on another page. Both column widths and headings may be changed within a report.

```
COLUMN SIZES  20, 4, 8, 8, 8
COLUMN HEADINGS "ACCOUNT", "UNIT", "1972", "1973", "1974"
```

The lines in a report would normally be single spaced, but the user may also request double or triple spacing over an entire report or over parts of a report. In addition, a series of lines may be skipped at any time with the statement

```
SKIP x LINES
```

where x is a number. A final method for controlling the spacing of lines is the use of a statement which causes the beginning of a new page. The form of the statement is simply

```
BEGIN NEW PAGE
```

The final type of statement in the report format specification language is the type used to specify the information to be printed on a line. The user can indicate a page title that is printed at the top of each page, a footing that is printed at the bottom of each page, a variable whose values are to be printed, or just a line of information that is to be printed.

A report is primarily produced in order to print the values calculated in a logic module. The stimulus for the printing of the values of a variable is given by an ITEM statement. The ITEM statement, such as the one below, specifies a description of the variable that is to be printed in the first column of the report, a description of the units of the values which is printed in the second column of the report, the number of decimal places the printed values are to have, and lastly the name of the variable as it appears in the logic module.

```
ITEM "COST OF GOODS SOLD", "(M$)", 0, CGS
```

The minimum information that must appear in an ITEM statement is the keyword "ITEM" and the name of the variable to be printed. If the description is not specified, the name of the variable is printed in the first column of the report. When no units are given, the second column of the report is left blank. In the event that the number of decimal places is not given, zero decimal places are assumed.

In order to specify that a line of information other than the values of variables is to be printed, a statement is used which consists of the word "LINE" in positions two through five with the information to be printed appearing in positions seven through seventy-two. Should more space be needed, the information may be continued in positions seven through seventy-two of the next line in the report module. Thus a maximum of 132 positions (2 times 72-7+1) is available for specifying a line of information.

Titles and footings that are printed at the top and bottom of each page, respectively, can be created using statements similar to the LINE statement. The only difference in form is that the words "TITLE" and "FOOT" are used instead of the word "LINE" and the body of a title cannot begin before the eighth position of the first line, giving a maximum title length of 131 characters.

A sample report module which demonstrates the use of the report format specification language is presented in Figure 4.

This chapter has described the structure of CMS/1 and has discussed the general design concepts and the capabilities of the system. The next chapter demonstrates the implementation of CMS/1 by presenting several models which depict the capabilities of the system.

CHAPTER IV

IMPLEMENTATION OF CMS/1

The previous chapters have introduced the concepts of corporate models and corporate modeling systems. The desired characteristics of corporate modeling systems were discussed and CMS/1, a new corporate modeling system, was introduced.

The present chapter demonstrates the capabilities of CMS/1 by demonstrating its use in two hypothetical situations - in the analysis of the potential profitability of a new product and in planning for a multi-divisional firm. The models developed for these hypothetical problems are relatively simple models. The intention is to demonstrate the use of CMS/1, not to develop sophisticated models.

The use of CMS/1 is demonstrated with the simple execution of a deterministic model, with sensitivity analyses, with a Monte Carlo simulation, and with the execution of a set of interrelated models.

Deterministic Model

A deterministic model for the analysis of the potential profitability of a new product was introduced previously in Figures 2, 3, and 4. The execution of that model would be initiated by the control statement

```
*EXECUTE  DATA  PRODUCT_DATA, REPORT NEW_PRODUCT
```

This statement would cause the data module "PRODUCT_DATA," the report module "NEW_PRODUCT," and the most recently created unnamed logic and data modules to be combined and executed. (The results of such an execution are depicted in Figure 5.)

Deterministic models are usually used to "try out" different alternatives. By changing data and/or logical relationships, various assumptions and "what if" questions can be investigated. For example, the model of the new product venture could be used to determine the effects of using different methods of depreciation.

The results of the sample model, depicted in Figure 5, are based on the use of straight-line depreciation of the original investment. The impact of different depreciation methods can be easily ascertained by altering the model. The simplest way to alter the model is to add an unnamed data module containing the new depreciation method. The following statements, when appended to the previous model, would create a data module which specifies the use of the sum-of-the-years-digits method of depreciation (method number 2), and then initiate the execution of the "new" model.

```
*DATA  PERIODS 1 TO 5
 SET DEPRECIATION EQUAL TO DEPRECIATE (2, INVESTMENT,
1    SALVAGE)
*EXECUTE  DATA PRODUCT_DATA, REPORT NEW_PRODUCT
```

A complete sequence of modules and EXECUTE statements for the comparison of straight-line, sum-of-the-years-digits, and double declining balance methods of depreciation are presented in Figure 6. The results for each of the three methods are presented in Figures 5, 7, and 8, respectively.

Sensitivity Analysis

As indicated above, a deterministic model is usually executed more than once in order to investigate various alternatives. Repetitive execution may also be used to determine the variables in a model which most affect the behavior of the model.

An example of the investigation of various alternatives was given in the previous section. In that case the investigator had to specify changes for each new alternative and then initiate a new execution. This procedure may be simplified when the objective is to determine the "sensitive" variables rather than to investigate new alternatives.

Sensitivity analysis is accomplished by repeatedly executing a model with the value of only one variable being changed for each new execution. Thus, if the multiple values can be specified for a variable before the repetitive executions begin, all of the executions may be carried out automatically. This is accomplished in CMS/1 by the use of the ITERATE function. The statement

```
  SET BEGINNING_MARKET_SIZE EQUAL TO ITERATE (180000,
 1    260000, 20000)
```

denotes that the variable BEGINNING_MARKET_SIZE is to have the value 180000 the first time the model is executed. The variable will then be incremented by 20000 in succeeding

Figure 6. Sample deterministic model.

```
*LOGIC
 SET MARKET_SIZE EQUAL TO GROWTH(BEGINNING_MARKET_SIZE, ,
1                 GROWTH_RATE)
 SET MARKET_SHARE EQUAL TO LINEAR(BEGINNING_SHARE,
1                 MAXIMUM_SHARE,)
 SET UNITS_SOLD EQUAL TO MARKET_SIZE TIMES MARKET_SHARE
 SET SALES EQUAL TO UNITS_SOLD TIMES PRICE
 SET VARIABLE_COSTS EQUAL TO VARIABLE_COST_RATE TIMES
1                 UNITS_SOLD
 SET TOTAL_COSTS EQUAL TO FIXED_COST PLUS VARIABLE_COSTS
1           PLUS DEPRECIATION
 SET OPERATING_INCOME EQUAL TO SALES MINUS TOTAL_COSTS
 SET TAX_EFFECT EQUAL TO TAX_RATE TIMES DEPRECIATION
 SET CASH_FLOW EQUAL TO OPERATING_INCOME PLUS SALVAGE
1           PLUS DEPRECIATION PLUS TAX_EFFECT
 SET PRESENT_VALUE EQUAL TO DISCOUNT(CASH_FLOW,
1                   DISCOUNT_RATE)
 SET PROFIT_INDEX EQUAL TO PRESENT_VALUE DIVIDED BY
1                 INVESTMENT
 SET ROI EQUAL TO INTRL_RATE(INVESTMENT, CASH_FLOW)
*DATA PRODUCT_DATA  PERIODS 1 TO 5
 SET INVESTMENT EQUAL TO 9600000, 0, 0, 0, 0
 SET SALVAGE EQUAL TO 0, 0, 0, 0, 4250000
 SET TAX_RATE EQUAL TO .48
 SET PRICE EQUAL TO 516
 SET FIXED_COST EQUAL TO 306350
 SET VARIABLE_COST_RATE EQUAL TO 421.65
 SET BEGINNING_MARKET_SIZE EQUAL TO 220000
 SET GROWTH_RATE EQUAL TO  , .08, .06, .03, .02
 SET BEGINNING_SHARE EQUAL TO .01
 SET MAXIMUM_SHARE EQUAL TO .15
 SET DISCOUNT_RATE EQUAL TO .12
 SET DEPRECIATION EQUAL TO DEPRECIATE(1,INVESTMENT,SALVAGE)
*REPORT  NEW_PRODUCT
 TITLE                         ANALYSIS OF NEW PRODUCT
 MARGIN  0
 COLUMN SIZES 15, 0, (9)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT",  "   1972",  "   1973",
1                 "   1974",  "   1975",  "   1976"
 COLUMN HEADINGS "-------", ("   ----")
 ITEM "INITIAL MARKET", BEGINNING_MARKET_SIZE
 ITEM 2, GROWTH_RATE
 ITEM 2, BEGINNING_SHARE
 ITEM 2, MAXIMUM_SHARE
 SKIP 1 LINE
```

```
 ITEM INVESTMENT
 ITEM DEPRECIATION
 ITEM "SALVAGE VALUE", SALVAGE
 SKIP 1 LINE
 ITEM MARKET_SIZE
 ITEM "SHARE OF MARKET", 3, MARKET_SHARE
 ITEM UNITS_SOLD
 SKIP 1 LINE
 ITEM "PRICE PER UNIT", 2, PRICE
 ITEM SALES
 ITEM VARIABLE_COSTS
 ITEM FIXED_COST
 ITEM TOTAL_COSTS
 ITEM  "NET INCOME", OPERATING_INCOME
 SKIP 1 LINE
 ITEM CASH_FLOW
 ITEM PRESENT_VALUE
 ITEM 2, PROFIT_INDEX
 ITEM 3, ROI
*EXECUTE  DATA PRODUCT_DATA, REPORT NEW_PRODUCT
*DATA PERIODS 1 TO 5
 SET DEPRECIATION EQUAL TO DEPRECIATE(2,INVESTMENT,SALVAGE)
*EXECUTE  DATA PRODUCT_DATA, REPORT NEW_PRODUCT
*DATA PERIODS 1 TO 5
 SET DEPRECIATION EQUAL TO DEPRECIATE(3, INVESTMENT, 5)
*EXECUTE  DATA PRODUCT_DATA, REPORT NEW_PRODUCT
```

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	1583333	1266666	950000	633333	316667
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2200	10692	20148	29832	39690
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5517067	10396593	15393451	20479968
VARIABLE COSTS	927630	4508277	8495586	12578773	16735228
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2817312	6081293	9751936	13518456	17358224
NET INCOME	-1682113	-564226	644657	1874995	3121744
CASH FLOW	661220	1310439	2050656	2812327	7840409
PRESENT VALUE	9330820	*	*	*	*
PROFIT INDEX	1.04	*	*	*	*
ROI	0.180	*	*	*	*

Figure 7. Results with sum-of-the-years digits depreciation.

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
-------	----	----	----	----	----
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	3599999	2159999	1295999	777600	466560
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2200	10692	20148	29832	39690
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5517067	10396593	15393451	20479968
VARIABLE COSTS	927630	4508277	8495586	12578773	16735228
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	4833978	6974626	10097935	13662722	17508112
NET INCOME	-3698779	-1457559	298658	1730729	2971856
CASH FLOW	1629219	1739239	2216736	2881575	7912363
PRESENT VALUE	10739990	*	*	*	*
PROFIT INDEX	1.19	*	*	*	*
ROI	0.259	*	*	*	*

Figure 8. Results with double declining balance depreciation.

executions of the model until a maximum value of 260000 is reached.

The ITERATE function specifies the values for a variable but does not control the number of times the model is executed. The number of repetitions is specified on the EXECUTE statement. The statement

```
*EXECUTE DATA PRODUCT_DATA, REPORT NEW_PRODUCT,
1    ITERATIONS 5
```

could be used in conjunction with the previous ITERATE function to ascertain the effects of varying the variable BEGINNING_MARKET_SIZE between the values 180000 and 260000.

Complete sequences of modules and EXECUTE statements which could be utilized in sensitivity analyses of the variables "BEGINNING_MARKET_SIZE" and "MAXIMUM_SHARE" are presented in Figure 9. (The assumption is made that the modules depicted in Figure 6 have been previously processed.) The results of each analysis are presented in Figures 10 and 11, respectively.

Monte Carlo Simulation

A deterministic model is useful in specifying the basic structure of a problem, in investigating alternative courses of action, and in determining the sensitive variables. But additional information concerning the riskiness of a course of action is often needed by management.

The deterministic model discussed above indicated that a return on investment of about 17 percent may be expected

```
*DATA   PERIODS   1 TO 5
 SET BEGINNING_MARKET_SIZE EQUAL TO ITERATE(180000, 260000,
1                         20000)
*EXECUTE    DATA PRODUCT_DATA,  REPORT  NEW_PRODUCT
1           ITERATIONS  5
*DATA   PERIODS   1 TO 5
 SET MAXIMUM_SHARE EQUAL TO ITERATE(.10, .18, .02)
*EXECUTE    DATA PRODUCT_DATA,  REPORT  NEW_PRODUCT
1           ITERATIONS  5
```

Figure 9. Addition to sample model for sensitivity analysis.

Figure 10. Results of sensitivity analysis for the initial market size.

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	180000	180000	180000	180000	180000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	180000	194400	206064	212246	216490
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	1800	8748	16485	24408	32474
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	928800	4513963	8506304	12594642	16756338
VARIABLE COSTS	758970	3688590	6950934	10291724	13692457
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2015319	4944940	8207284	11548074	14948807
NET INCOME	-1086519	-430977	299020	1046568	1807531
CASH FLOW	319481	975023	1705019	2452567	7463530
PRESENT VALUE	8069800	*	*	*	*
PROFIT INDEX	0.90	*	*	*	*
ROI	0.122	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	200000	200000	200000	200000	200000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	200000	216000	228960	235829	240545
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2000	9720	18317	27120	36082
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1032000	5015515	9451451	13994049	18618144
VARIABLE COSTS	843300	4098433	7723262	11435250	15213846
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2099649	5354783	8979612	12691600	16470196
NET INCOME	-1067649	-339268	471839	1302449	2147948
CASH FLOW	338351	1066731	1877838	2708448	7803947
PRESENT VALUE	8638545	*	*	*	*
PROFIT INDEX	0.96	*	*	*	*
ROI	0.147	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2200	10692	20148	29832	39690
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5517067	10396593	15393451	20479968
VARIABLE COSTS	927630	4508277	8495586	12578773	16735228
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	5764627	9751936	13835123	17991568
NET INCOME	-1048780	-247560	644657	1558328	2488400
CASH FLOW	357220	1158439	2050656	2964327	8144399
PRESENT VALUE	9207309	*	*	*	*
PROFIT INDEX	1.02	*	*	*	*
ROI	0.172	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	240000	240000	240000	240000	240000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	240000	259200	274752	282994	288654
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2400	11664	21980	32544	43298
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1238399	6018617	11341740	16792848	22341776
VARIABLE COSTS	1011959	4918119	9267914	13722299	18256608
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2268309	6174469	10524264	14978649	19512944
NET INCOME	-1029910	-155852	817476	1814199	2828832
CASH FLOW	376090	1250147	2223475	3220198	8484831
PRESENT VALUE	9776057	*	*	*	*
PROFIT INDEX	1.09	*	*	*	*
ROI	0.197	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	260000	260000	260000	260000	260000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.15	0.15	0.15	0.15	0.15
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	260000	280800	297648	306577	312708
SHARE OF MARKET	0.010	0.045	0.080	0.115	0.150
UNITS SOLD	2600	12636	23812	35256	46906
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1341599	6520169	12286887	18192256	24203600
VARIABLE COSTS	1096289	5327963	10040241	14865827	19778000
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2352639	6584313	11296591	16122177	21034336
NET INCOME	-1011040	-64144	990296	2070079	3169264
CASH FLOW	394960	1341855	2396295	3476078	8825263
PRESENT VALUE	10344810	*	*	*	*
PROFIT INDEX	1.15	*	*	*	*
ROI	0.221	*	*	*	*

Figure 11. Results of sensitivity analysis for the maximum market share.

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.10	0.10	0.10	0.10	0.10
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251855	259411	264599
SHARE OF MARKET	0.010	0.032	0.055	0.077	0.100
UNITS SOLD	2200	7722	13852	20104	26460
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	3984547	7147660	10373847	13653313
VARIABLE COSTS	927630	3255977	5840717	8476999	11156818
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	4512327	7097067	9733349	12413168
NET INCOME	-1048780	-527780	50593	640498	1240145
CASH FLOW	357220	878220	1456592	2046497	6896144
PRESENT VALUE	7269483	*	*	*	*
PROFIT INDEX	0.81	*	*	*	*
ROI	0.084	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.12	0.12	0.12	0.12	0.12
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.037	0.065	0.092	0.120
UNITS SOLD	2200	8910	16371	23996	31752
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	4597553	8447232	12381690	16383977
VARIABLE COSTS	927630	3756895	6902664	10117710	13388183
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	5013245	8159014	11374060	14644533
NET INCOME	-1048780	-415692	288218	1007630	1739444
CASH FLOW	357220	990308	1694217	2413629	7395443
PRESENT VALUE	8044612	*	*	*	*
PROFIT INDEX	0.89	*	*	*	*
ROI	0.121	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
-------	----	----	----	----	----
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.14	0.14	0.14	0.14	0.14
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.042	0.075	0.107	0.140
UNITS SOLD	2200	10098	18889	27887	37044
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5210561	9746806	14389530	19114624
VARIABLE COSTS	927630	4257815	7964612	11758419	15619546
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	5514165	9220962	13014769	16875888
NET INCOME	-1048780	-303604	525844	1374761	2238736
CASH FLOW	357220	1102395	1931843	2780760	7894735
PRESENT VALUE	8819736	*	*	*	*
PROFIT INDEX	0.98	*	*	*	*
ROI	0.155	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.16	0.16	0.16	0.16	0.16
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.047	0.085	0.122	0.160
UNITS SOLD	2200	11286	21408	31778	42336
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	5823567	11046380	16397373	21845296
VARIABLE COSTS	927630	4758734	9026560	13399129	17850896
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	6015084	10282910	14655479	19107232
NET INCOME	-1048780	-191517	763470	1741894	2738064
CASH FLOW	357220	1214482	2169469	3147893	8394063
PRESENT VALUE	9594882	*	*	*	*
PROFIT INDEX	1.07	*	*	*	*
ROI	0.189	*	*	*	*

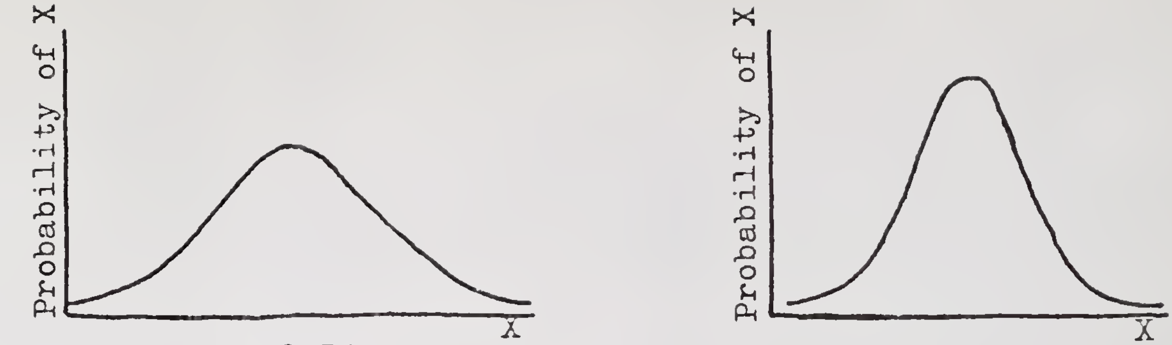
ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
INITIAL MARKET	220000	220000	220000	220000	220000
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.18	0.18	0.18	0.18	0.18
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	220000	237600	251856	259411	264599
SHARE OF MARKET	0.010	0.052	0.095	0.137	0.180
UNITS SOLD	2200	12474	23926	35669	47628
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1135199	6436577	12345955	18405200	24575952
VARIABLE COSTS	927630	5259655	10088509	15039838	20082272
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2183979	6516005	11344859	16296188	21338608
NET INCOME	-1048780	-79428	1001096	2109012	3237344
CASH FLOW	357220	1326571	2407095	3515011	8893343
PRESENT VALUE	10369990	*	*	*	*
PROFIT INDEX	1.15	*	*	*	*
ROI	0.220	*	*	*	*

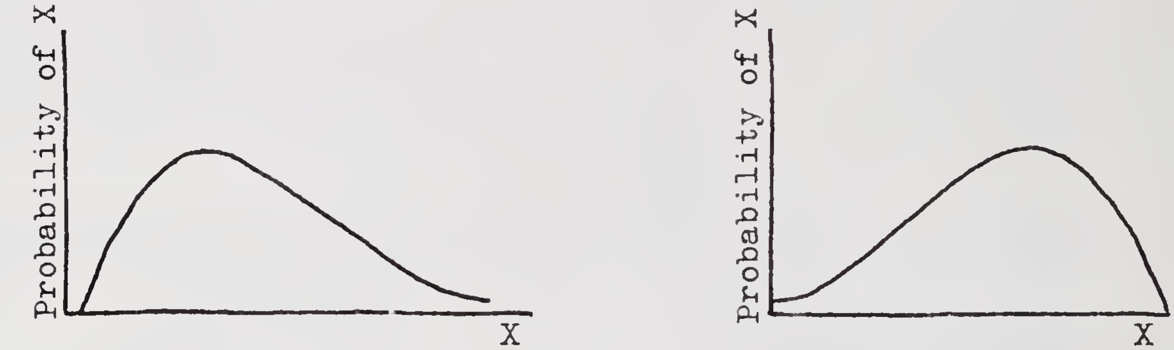
from the hypothetical new product venture (see Figure 5). But no information was provided concerning the riskiness of the venture. Can the venture lose money? What is the minimum return that may be expected? What is the maximum? These questions concerning the probability of occurence of the various possible outcomes cannot be answered by deterministic models.

This type of information can be derived, however, from Monte Carlo simulations. Monte Carlo simulation requires that the modeling system be capable of dealing with random variables and be capable of executing a model repetitively.

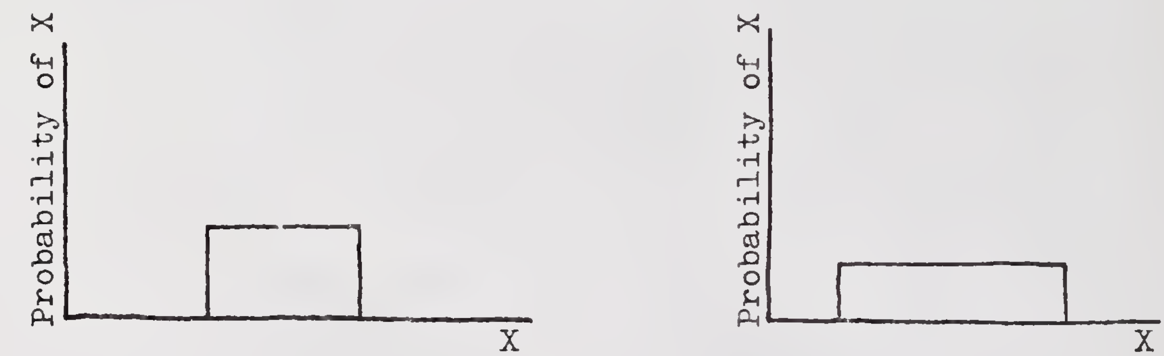
CMS/1 can accept four types of random variables. A random variable may conform to a normal distribution, a Weibull distribution, a uniform distribution, or an arbitrary distribution expressed as a relative frequency distribution (see Figure 12). The normal distribution may often be used in describing empirical data, particularly the distribution of averages such as an average per unit cost. The Weibull distribution has the useful characteristic of being bounded on one side. This is advantageous for expressing variables such as percentages as random variables. In that particular case a Weibull distribution with an upper bound of 100 percent could be used. If a normal, Weibull, or uniform distribution is not appropriate, a distribution may be expressed as a relative frequency distribution. This capability is particularly useful in describing empirical data which may not conform to any known mathematical distribution; e.g., the age distribution of accounts



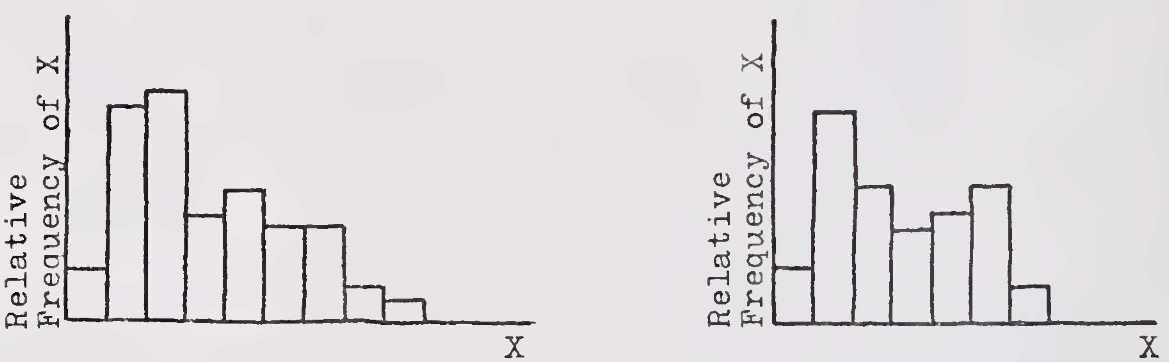
A. Normal Distributions



B. Weibull Distributions



C. Uniform Distributions



D. Relative Frequency Distributions

Figure 12. Types of distributions accepted by CMS/1.

receivable or a distribution of the costs associated with an operation.

In order to perform a Monte Carlo simulation, the number of times the model is to be executed must be stated. This is accomplished by the same procedure used in sensitivity analysis. That is, the parameter "ITERATIONS" is stipulated on the EXECUTE statement.

The sample deterministic model presented in Figure 6 may be altered for execution in a Monte Carlo simulation with the addition of the modules depicted in Figure 13. The data module in Figure 13 alters three variables to make them random variables. In addition, the report module named "RISK" is added in order to print out the new information supplied by the Monte Carlo simulation. The results of the simulation are presented in Figure 14.

The values in Figure 14 that are printed in the format specified by the report "NEW_PRODUCT" are the average values of the variables for the two hundred iterations. The report "RISK" displays the distributions of the values of two selected variables (i.e., CASH_FLOW and ROI). The distributions indicate that the average value of the cash flow increases over the years but the distribution of possible values "spreads out." Also, while the expected return on investment is high (29.8%), there is more than a 2.5 percent chance that the rate of return will be negative and approximately one chance in seven that the rate of return will be less that 11.5 percent. The Monte Carlo simulation thus provides information concerning the risk involved in the

```
*DATA  PERIODS  1 TO 5
 SET BEGINNING_MARKET_SIZE EQUAL TO NORMAL(180000, 260000)
 SET MAXIMUM_SHARE EQUAL TO WEIBULL(.20, .15, .06)
     VARIABLE_COST_RATE IS DISTRIBUTED(360,.1, 375,.1,
1              390,.11, 405,.12, 420,.13, 435,.13, 450,.11,
1              465,.08, 480,.06, 495,.04, 510,.02)
*REPORT  RISK
 TITLE                           ANALYSIS OF NEW PRODUCT
 MARGIN  0
 COLUMN SIZES 15, 0, (9)
 BEGIN NEW PAGE
 SINGLE SPACE
 DISTRIBUTION
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT",  "   1972",  "   1973",
1                 "   1974",  "   1975",  "   1976"
 COLUMN HEADINGS "--------", ("   ----")
 ITEM  CASH_FLOW
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT",  "   1972",  "   1973",
1                 "   1974",  "   1975",  "   1976"
 COLUMN HEADINGS "--------", ("   ----")
 ITEM  3, ROI
*EXECUTE  DATA PRODUCT_DATA,  REPORT  NEW_PRODUCT, RISK
1         ITERATIONS 200
```

Figure 13. Additions to sample model for Monte Carlo simulation.

Figure 14. Results of Monte Carlo simulation.

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
-------	----	----	----	----	----
INITIAL MARKET	219500	219500	219500	219500	219500
GROWTH RATE	*	0.08	0.06	0.03	0.02
BEGINNING SHARE	0.01	0.01	0.01	0.01	0.01
MAXIMUM SHARE	0.23	0.23	0.23	0.23	0.23
INVESTMENT	9000000	0	0	0	0
DEPRECIATION	950000	950000	950000	950000	950000
SALVAGE VALUE	0	0	0	0	4250000
MARKET SIZE	219500	237061	251284	258823	263999
SHARE OF MARKET	0.010	0.066	0.122	0.178	0.234
UNITS SOLD	2195	15618	30597	45979	61652
PRICE PER UNIT	516.00	516.00	516.00	516.00	516.00
SALES	1132641	8058922	15788382	23725136	31811712
VARIABLE COSTS	921659	6558469	12848931	19308256	25889392
FIXED COST	306350	306350	306350	306350	306350
TOTAL COSTS	2177963	7814827	14105287	20564608	27145776
NET INCOME	-1045362	244102	1683051	3160625	4666165
CASH FLOW	360624	1650076	3089004	4566593	10322124
PRESENT VALUE	12595430	*	*	*	*
PROFIT INDEX	1.40	*	*	*	*
ROI	0.298	*	*	*	*

ANALYSIS OF NEW PRODUCT

ACCOUNT	1972	1973	1974	1975	1976
CASH FLOW	0.015				
186161					
	0.985	0.215	0.045	0.020	
1038575					
		0.395	0.170	0.055	
1890988					
		0.335	0.195	0.140	
2743401					
		0.050	0.210	0.095	
3595814					
		0.005	0.245	0.180	
4448227					
			0.080	0.125	0.015
5300640					
			0.050	0.180	0.030
6153053					
				0.100	0.055
7005466					
			0.005	0.045	0.115
7857879					
				0.040	0.065
8710292					
				0.015	0.120
9562705					
					0.115
10415118					
				0.005	0.095
11267531					
					0.160
12119944					
					0.085
12972357					
					0.050
13824770					
					0.035
14677183					
					0.040
15529596					
					0.020
MEAN VALUE	360624	1650076	3089004	4566593	10322124
STD. DEV.	87232	652111	1286853	1939049	2603743

ANALYSIS OF NEW PRODUCT

ACCOUNT		1972	1973	1974	1975	1976
ROI		0.025				
	-0.031					
		0.020				
	0.006					
		0.020				
	0.042					
		0.035				
	0.079					
		0.040				
	0.115					
		0.080				
	0.152					
		0.055				
	0.188					
		0.035				
	0.225					
		0.100				
	0.261					
		0.075				
	0.298					
		0.065				
	0.334					
		0.070				
	0.371					
		0.120				
	0.407					
		0.065				
	0.444					
		0.075				
	0.480					
		0.035				
	0.517					
		0.025				
	0.553					
		0.035				
	0.589					
		0.020				
	0.626					
		0.005				
MEAN VALUE		0.298	0.000	0.000	0.000	0.000
STD. DEV.		0.164	0.000	0.000	0.000	0.000

new investment.

Interrelated Models

The use of CMS/1 in constructing interrelated models is demonstrated in this section by the development of a model which produces a corporate income statement. The corporate income statement is developed from information in divisional income statements which in turn receive information from income statements for the individual product lines. Thus, models for the products, divisions, and for the total corporation must be developed and interrelated. The corporation will be assumed to be composed of two divisions each producing two products. Since the products' income statements are needed before the divisional and corporate statements can be prepared, the product model will be discussed first.

The model for each product is composed of common logic and report modules (see Figure 15) combined with a different data module for each product. The logic module contains relationships which compute the revenue, the costs, and the contribution of each product. The computed information is then saved by the use of a SAVE statement so that it can be referenced by the divisional models.

A sequence of data modules and EXECUTE statements for the firm's four products is presented in Figure 16. The output from the execution of the four models is given in Figures 17 through 20.

After the product models have been run, the income

```
*LOGIC   PRODUCT
 GROUP   OVERHEAD, R_AND_D UNDER FIXED_PRODUCT_EXPENSES
 GROUP   MATERIALS, LABOR, OVERHEAD, SELLING_COSTS UNDER
1        COST_RATES, VARIABLE_COSTS
 SALES EQUAL  UNIT_SALES TIMES PRICE
 VARIABLE_COSTS = UNIT_SALES * COST_RATES
 MARGINAL_COST EQUALS SUM OF VARIABLE_COSTS
 MARGINAL_INCOME = SALES MINUS MARGINAL_COST
 TOTAL_FIXED_EXPENSES = SUM OF FIXED_PRODUCT_EXPENSES
 PRODUCT_CONTRIBUTION = MARGINAL_INCOME -
1           TOTAL_FIXED_EXPENSES
 SAVE   SALES"PRODUCT_NO, MARGINAL_COST"PRODUCT_NO,
1       TOTAL_FIXED_EXPENSES"PRODUCT_NO,
1       PRODUCT_CONTRIBUTION"PRODUCT_NO
*REPORT  PRODUCT
 TITLE                        PRODUCT QUARTERLY PLAN
 MARGIN 0
 LINE LENGTH 60
 COLUMN SIZES  26, 0, (8)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT", " FIRST", " SECOND",
1                 " THIRD", " FOURTH"
 COLUMN HEADINGS  "--------", " -----", " ------",
1                 " -----", " ------"
 ITEM  SALES
 SKIP 1 LINE
 ITEM "DIRECT MATERIALS", VARIABLE_COSTS:MATERIALS
 ITEM "DIRECT LABOR", VARIABLE_COSTS:LABOR
 ITEM "VARIABLE MFG. OVERHEAD", VARIABLE_COSTS:OVERHEAD
 ITEM "VARIABLE SELLING EXPENSES", VARIABLE_COSTS:
1         SELLING_COSTS
 ITEM "TOTAL MARGINAL COSTS", MARGINAL_COST
 SKIP 1 LINE
 ITEM MARGINAL_INCOME
 SKIP 1 LINE
 ITEM "FIXED MFG. OVERHEAD", FIXED_PRODUCT_EXPENSES:OVERHEAD
 ITEM "R & D EXPENSE", FIXED_PRODUCT_EXPENSES:R_AND_D
 SKIP 1 LINE
 ITEM PRODUCT_CONTRIBUTION
```

Figure 15. Logic and report modules for products.

Figure 16. Data modules and execute statements for products.

```
*DATA  PERIODS 1 TO 4
 GROUP  OVERHEAD, R_AND_D UNDER FIXED_PRODUCT_EXPENSES
 GROUP  MATERIALS, LABOR, OVERHEAD, SELLING_COSTS UNDER
1       COST_RATES
 PRODUCT_NO IS 1
 UNIT_SALES ARE  10000, 18000, 16000, 10000
 PRICE IS  150
 COST_RATES:MATERIALS = 35
 COST_RATES:LABOR = 40
 COST_RATES:OVERHEAD = 10
 COST_RATES:SELLING_COSTS = 8
 FIXED_PRODUCT_EXPENSES:OVERHEAD = 300000
 FIXED_PRODUCT_EXPENSES:R_AND_D = 100000
*EXECUTE  LOGIC PRODUCT,  SAVE PRODUCT_11
1          REPORT PRODUCT,  HEADING "DIVISION 1 - PRODUCT 1"
*DATA  PERIODS 1 TO 4
 GROUP  OVERHEAD, R_AND_D UNDER FIXED_PRODUCT_EXPENSES
 GROUP  MATERIALS, LABOR, OVERHEAD, SELLING_COSTS UNDER
1       COST_RATES
 PRODUCT_NO IS 2
 UNIT_SALES ARE  30000, 20000, 16000, 28000
 PRICE IS  120
 COST_RATES:MATERIALS = 25
 COST_RATES:LABOR = 35
 COST_RATES:OVERHEAD = 10
 COST_RATES:SELLING_COSTS = 6
 FIXED_PRODUCT_EXPENSES:OVERHEAD = 350000
 FIXED_PRODUCT_EXPENSES:R_AND_D =  50000
*EXECUTE  LOGIC PRODUCT,  SAVE PRODUCT_12
1          REPORT PRODUCT,  HEADING "DIVISION 1 - PRODUCT 2"
```

```
*DATA  PERIODS 1 TO 4
 GROUP  OVERHEAD, R_AND_D UNDER FIXED_PRODUCT_EXPENSES
 GROUP  MATERIALS, LABOR, OVERHEAD, SELLING_COSTS UNDER
1        COST_RATES
 PRODUCT_NO IS 1
 UNIT_SALES ARE  15000, 20000, 18000, 14000
 PRICE IS  130
 COST_RATES:MATERIALS = 30
 COST_RATES:LABOR = 40
 COST_RATES:OVERHEAD = 9
 COST_RATES:SELLING_COSTS = 6
 FIXED_PRODUCT_EXPENSES:OVERHEAD = 300000
 FIXED_PRODUCT_EXPENSES:R_AND_D =  75000
*EXECUTE  LOGIC PRODUCT,  SAVE PRODUCT_21
1          REPORT PRODUCT,  HEADING "DIVISION 2 - PRODUCT 1"
*DATA  PERIODS 1 TO 4
 GROUP  OVERHEAD, R_AND_D UNDER FIXED_PRODUCT_EXPENSES
 GROUP  MATERIALS, LABOR, OVERHEAD, SELLING_COSTS UNDER
1        COST_RATES
 PRODUCT_NO IS 2
 UNIT_SALES ARE  40000, 28000, 26000, 36000
 PRICE IS  90
 COST_RATES:MATERIALS = 25
 COST_RATES:LABOR = 30
 COST_RATES:OVERHEAD = 9
 COST_RATES:SELLING_COSTS = 5
 FIXED_PRODUCT_EXPENSES:OVERHEAD = 250000
 FIXED_PRODUCT_EXPENSES:R_AND_D =  50000
*EXECUTE  LOGIC PRODUCT,  SAVE PRODUCT_22
1          REPORT PRODUCT,  HEADING "DIVISION 2 - PRODUCT 2"
```

DIVISION 1 - PRODUCT 1

PRODUCT QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	1500000	2700000	2400000	1500000
DIRECT MATERIALS	350000	630000	560000	350000
DIRECT LABOR	400000	720000	640000	400000
VARIABLE MFG. OVERHEAD	100000	180000	160000	100000
VARIABLE SELLING EXPENSES	80000	144000	128000	80000
TOTAL MARGINAL COSTS	930000	1674000	1488000	930000
MARGINAL INCOME	570000	1026000	912000	570000
FIXED MFG. OVERHEAD	300000	300000	300000	300000
R & D EXPENSE	100000	100000	100000	100000
PRODUCT CONTRIBUTION	170000	626000	512000	170000

Figure 17. Model results for Division One, Product One.

DIVISION 1 - PRODUCT 2

PRODUCT QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	3600000	2400000	1920000	3360000
DIRECT MATERIALS	750000	500000	400000	700000
DIRECT LABOR	1050000	700000	560000	980000
VARIABLE MFG. OVERHEAD	300000	200000	160000	280000
VARIABLE SELLING EXPENSES	180000	120000	96000	168000
TOTAL MARGINAL COSTS	2280000	1520000	1216000	2128000
MARGINAL INCOME	1320000	880000	704000	1232000
FIXED MFG. OVERHEAD	350000	350000	350000	350000
R & D EXPENSE	50000	50000	50000	50000
PRODUCT CONTRIBUTION	920000	480000	304000	832000

Figure 18. Model results for Division One, Product Two.

DIVISION 2 - PRODUCT 1

PRODUCT QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	1950000	2600000	2340000	1820000
DIRECT MATERIALS	450000	600000	540000	420000
DIRECT LABOR	600000	800000	720000	560000
VARIABLE MFG. OVERHEAD	135000	180000	162000	126000
VARIABLE SELLING EXPENSES	90000	120000	108000	84000
TOTAL MARGINAL COSTS	1275000	1700000	1530000	1190000
MARGINAL INCOME	675000	900000	810000	630000
FIXED MFG. OVERHEAD	300000	300000	300000	300000
R & D EXPENSE	75000	75000	75000	75000
PRODUCT CONTRIBUTION	300000	525000	435000	255000

Figure 19. Model results for Division Two, Product One.

DIVISION 2 - PRODUCT 2

PRODUCT QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	3600000	2520000	2340000	3240000
DIRECT MATERIALS	1000000	700000	650000	900000
DIRECT LABOR	1200000	840000	780000	1080000
VARIABLE MFG. OVERHEAD	360000	252000	234000	324000
VARIABLE SELLING EXPENSES	200000	140000	130000	180000
TOTAL MARGINAL COSTS	2760000	1932000	1794000	2484000
MARGINAL INCOME	840000	588000	546000	756000
FIXED MFG. OVERHEAD	250000	250000	250000	250000
R & D EXPENSE	50000	50000	50000	50000
PRODUCT CONTRIBUTION	540000	288000	246000	456000

Figure 20. Model results for Division Two, Product Two.

statements for the two divisions can be computed. Again, common logic and report modules are used in combination with different data modules for each division (see Figure 21). The division models combine information concerning the two products of each division with fixed expenses incurred at the divisional level to give the net contribution gained from each division. The summarized information is then saved with a SAVE statement for later use in the corporate model.

The data modules and EXECUTE statements used in the execution of the divisional models are presented in Figure 22. The income statements for the two divisions are depicted in Figures 23 and 24.

The final step in the execution of the interrelated models is to combine the information saved from the divisional models with data on corporate expenses in order to compute the corporate income. The complete corporate model, i.e., logic, data, and report modules, is presented in Figure 25. No information is saved by the corporate logic since no other model uses the information computed.

The final corporate income statement which has been "built up" from the product level is given in Figure 26.

```
*LOGIC  DIVISION
 GROUP  MFG_OVERHEAD, R_AND_D, SELLING_AND_ADVERTISING,
1       ADMINISTRATION  UNDER DIVISION_EXPENSES
 SALES EQUAL SALES1 PLUS SALES2
 MARGINAL_COST = MARGINAL_COST1 + MARGINAL_COST2
 FIXED_PRODUCT_EXPENSES = TOTAL_FIXED_EXPENSES1 PLUS
1                         TOTAL_FIXED_EXPENSES2
 TOTAL_PRODUCT_COSTS = MARGINAL_COST +
1          FIXED_PRODUCT_EXPENSES
 CONTRIBUTION_FROM_PRODUCTS = SALES - TOTAL_PRODUCT_COSTS
 TOTAL_DIVISION_EXPENSES = SUM OF DIVISION_EXPENSES
 CONTRIBUTION_FROM_DIVISION = CONTRIBUTION_FROM_PRODUCTS -
1          TOTAL_DIVISION_EXPENSES
 SAVE  SALES"DIVISION_NO, TOTAL_PRODUCT_COSTS"DIVISION_NO,
1      TOTAL_DIVISION_EXPENSES"DIVISION_NO,
1      CONTRIBUTION_FROM_DIVISION"DIVISION_NO
*REPORT  DIVISION
 TITLE                    DIVISIONAL QUARTERLY PLAN
 MARGIN 0
 LINE LENGTH 60
 COLUMN SIZES  26, 0, (8)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT", " FIRST", " SECOND",
1                 " THIRD", " FOURTH"
 COLUMN HEADINGS  "-------", " -----", " ------",
1                 " -----", " ------"
 ITEM SALES
 SKIP 1 LINE
 LINE  PRODUCT COSTS:
 ITEM MARGINAL_COST
 ITEM "FIXED COST", FIXED_PRODUCT_EXPENSES
 ITEM CONTRIBUTION_FROM_PRODUCTS
 SKIP 1 LINE
 LINE  DIVISIONAL COSTS:
 ITEM "FIXED MFG. OVERHEAD", DIVISION_EXPENSES:MFG_OVERHEAD
 ITEM "R & D EXPENSES", DIVISION_EXPENSES:R_AND_D
 ITEM DIVISION_EXPENSES:SELLING_AND_ADVERTISING
 ITEM DIVISION_EXPENSES:ADMINISTRATION
 ITEM TOTAL_DIVISION_EXPENSES
 SKIP 1 LINE
 ITEM CONTRIBUTION_FROM_DIVISION
```

Figure 21. Logic and report modules for divisions.

```
*DATA   PERIODS 1 TO 4
 GROUP   MFG_OVERHEAD, R_AND_D, SELLING_AND_ADVERTISING,
1        ADMINISTRATION  UNDER DIVISION_EXPENSES
 DIVISION_NO IS 1
 DIVISION_EXPENSES:MFG_OVERHEAD = 200000
 DIVISION_EXPENSES:R_AND_D =  50000
 DIVISION_EXPENSES:SELLING_AND_ADVERTISING = 250000
 DIVISION_EXPENSES:ADMINISTRATION = 150000
*EXECUTE  LOGIC DIVISION, DATA PRODUCT_11 PRODUCT_12,
1          SAVE DIVISION_1
1          REPORT DIVISION, HEADING "DIVISION 1"
*DATA   PERIODS 1 TO 4
 GROUP   MFG_OVERHEAD, R_AND_D, SELLING_AND_ADVERTISING,
1        ADMINISTRATION  UNDER DIVISION_EXPENSES
 DIVISION_NO IS 2
 DIVISION_EXPENSES:MFG_OVERHEAD = 100000
 DIVISION_EXPENSES:R_AND_D =  50000
 DIVISION_EXPENSES:SELLING_AND_ADVERTISING = 200000
 DIVISION_EXPENSES:ADMINISTRATION = 150000
*EXECUTE  LOGIC DIVISION, DATA PRODUCT_21 PRODUCT_22,
1          SAVE DIVISION_2
1          REPORT DIVISION, HEADING "DIVISION 2"
```

Figure 22. Data modules and execute statements for divisions.

DIVISION 1

DIVISIONAL QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	5100000	5100000	4320000	4860000
PRODUCT COSTS:				
MARGINAL COST	3210000	3194000	2704000	3058000
FIXED COST	800000	800000	800000	800000
CONTRIBUTION FROM PRODUCTS	1090000	1106000	816000	1002000
DIVISIONAL COSTS:				
FIXED MFG. OVERHEAD	200000	200000	200000	200000
R & D EXPENSES	50000	50000	50000	50000
SELLING AND ADVERTISING	250000	250000	250000	250000
ADMINISTRATION	150000	150000	150000	150000
TOTAL DIVISION EXPENSES	650000	650000	650000	650000
CONTRIBUTION FROM DIVISION	440000	456000	166000	352000

Figure 23. Model results for Division One.

DIVISION 2

DIVISIONAL QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	5550000	5120000	4680000	5060000
PRODUCT COSTS:				
MARGINAL COST	4035000	3632000	3324000	3674000
FIXED COST	675000	675000	675000	675000
CONTRIBUTION FROM PRODUCTS	840000	813000	681000	711000
DIVISIONAL COSTS:				
FIXED MFG. OVERHEAD	100000	100000	100000	100000
R & D EXPENSES	50000	50000	50000	50000
SELLING AND ADVERTISING	200000	200000	200000	200000
ADMINISTRATION	150000	150000	150000	150000
TOTAL DIVISION EXPENSES	500000	500000	500000	500000
CONTRIBUTION FROM DIVISION	340000	313000	181000	211000

Figure 24. Model results for Division Two.

Figure 25. Complete corporate level model.

```
*LOGIC  CORPORATION
 GROUP  ADVERTISING, ADMINISTRATION, OTHER UNDER
1       CORPORATE_EXPENSES
 SALES = SALES1 + SALES2
 PRODUCT_COSTS = TOTAL_PRODUCT_COSTS1 PLUS
1                TOTAL_PRODUCT_COSTS2
 DIVISION_EXPENSES = TOTAL_DIVISION_EXPENSES1 PLUS
1                    TOTAL_DIVISION_EXPENSES2
 CONTRIBUTION_FROM_DIVISIONS = CONTRIBUTION_FROM_DIVISION1
1                             PLUS CONTRIBUTION_FROM_DIVISION2
 TOTAL_CORPORATE_EXPENSES = SUM OF CORPORATE_EXPENSES
 NET_INCOME_BEFORE_TAXES = CONTRIBUTION_FROM_DIVISIONS -
1         TOTAL_CORPORATE_EXPENSES
 IF NET_INCOME_BEFORE_TAXES < 0  THEN JUMP TO NO_TAX
 IF NET_INCOME_BEFORE_TAXES > 25000 THEN TAXES = .48 TIMES
1       NET_INCOME_BEFORE_TAXES - 6500
1                                         ELSE TAXES = .22 TIMES
1       NET_INCOME_BEFORE_TAXES
 JUMP TO AFTER_TAX
 NO_TAX) TAXES = 0
 AFTER_TAX)
 NET_INCOME_AFTER_TAXES = NET_INCOME_BEFORE_TAXES - TAXES
*REPORT  CORPORATION
 TITLE                        CORPORATE QUARTERLY PLAN
 MARGIN 0
 LINE LENGTH 60
 COLUMN SIZES  24, 0, (9)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS  "ACCOUNT", " FIRST", " SECOND",
1                 " THIRD", "  FOURTH"
 COLUMN HEADINGS  "-------", " -----", " ------",
1                 " -----", "  ------"
 ITEM SALES
 SKIP 1 LINE
 ITEM PRODUCT_COSTS
 ITEM DIVISION_EXPENSES
 LINE  CONTRIBUTION FROM
 ITEM " DIVISIONS", CONTRIBUTION_FROM_DIVISIONS
 SKIP 1 LINE
```

```
 LINE  CORPORATE EXPENSES:
 ITEM CORPORATE_EXPENSES:ADVERTISING
 ITEM CORPORATE_EXPENSES:ADMINISTRATION
 ITEM CORPORATE_EXPENSES:OTHER
 ITEM TOTAL_CORPORATE_EXPENSES
 SKIP 1 LINE
 LINE  NET INCOME
 ITEM "  BEFORE TAXES", NET_INCOME_BEFORE_TAXES
 SKIP 1 LINE
 ITEM TAXES
 SKIP 1 LINE
 LINE  NET INCOME
 ITEM "  AFTER TAXES", NET_INCOME_AFTER_TAXES
*DATA  PERIODS 1 TO 4
 GROUP  ADVERTISING, ADMINISTRATION, OTHER UNDER
1       CORPORATE_EXPENSES
 CORPORATE_EXPENSES:ADVERTISING = 100000
 CORPORATE_EXPENSES:ADMINISTRATION = 150000
 CORPORATE_EXPENSES:OTHER =  50000
*EXECUTE  LOGIC CORPORATION, DATA DIVISION_1 DIVISION_2
1          REPORT CORPORATION
```

CORPORATE QUARTERLY PLAN

ACCOUNT	FIRST	SECOND	THIRD	FOURTH
SALES	10650000	10220000	9000000	9920000
PRODUCT COSTS	8720000	8301000	7503000	8207000
DIVISION EXPENSES	1150000	1150000	1150000	1150000
CONTRIBUTION FROM DIVISIONS	780000	769000	347000	563000
CORPORATE EXPENSES:				
ADVERTISING	100000	100000	100000	100000
ADMINISTRATION	150000	150000	150000	150000
OTHER	50000	50000	50000	50000
TOTAL CORPORATE EXPENSES	300000	300000	300000	300000
NET INCOME BEFORE TAXES	480000	469000	47000	263000
TAXES	223900	218620	16060	119740
NET INCOME AFTER TAXES	256100	250380	30940	143260

Figure 26. Results for corporate level model.

CHAPTER V

SUMMARY AND RECOMMENDATIONS

Summary

This study was begun on the basic premise that a computer-based corporate modeling system could have the sophisticated capabilities required for corporate modeling while retaining a high level of management orientation.

Current systems and previous studies were examined in order to determine the desirable attributes of corporate modeling systems and the degree of management orientation which had already been attained. A new corporate modeling system was then developed in an attempt to improve the interface between corporate modelers and computers.

The modeling system, CMS/1, is composed of three "English-like" languages. These languages simplify the creation of logic, data, and report modules which are combined to form a corporate model. In addition, the modeling system has several advanced capabilities such as the ability to transfer the values of variables among models and the ability to perform sensitivity and risk analyses.

CMS/1 is designed in accordance with four general concepts. The most important concept is that the facilities supplied by the system should be determined by the requirements for corporate modeling rather than by the criteria of ease of implementation.

The second consideration in the design of CMS/1 is that the system should be convenient to use. This is promoted by the use of "English-like" languages that have short-forms which the experienced modeler may use. CMS/1 also assists the user through extensive error checking capabilities and by assuming a value for any system parameter which the user leaves unspecified.

A third design characteristic is that the modeling system should be applicable to different firms and to different uses. For example, modeling systems can be used in the analysis of special problems and in budgeting as well as in planning. This flexibility is primarily provided by CMS/1 through the capability of dealing with varying time horizons, the referencing of all variables by names rather than by positions, and the creation of models from modules which are developed separately and stored in a computer accessible form.

A final concept which guided the design of CMS/1 is that the modeling system itself should not have severe limitations in regard to the capacity of the system. Consequently, dynamic storage allocation is used to "create" tables for the storage of values as they are needed rather than using predefined tables with fixed capacities.

Recommendations for Further Research

The recommendations for further research may be grouped into two categories - (1) the extention of the basic capabilities of CMS/1 and (2) the alteration of CMS/1 to operate in an interactive mode.

The first recommendation is that the basic capabilities of CMS/1 may be extended. The current system can perform the basic arithmetic operations and special functions such as the depreciation of assets and the computation of present values and rates of return. Additional facilities could be added for such computations as the retirement of assets or the use of learning curves. Also, the system could be extended by improving the data handling capabilities. For example, under the current system, a data module cannot be easily altered. If data for a new period are to be added or an old period is to be deleted, the module must be reconstructed. It would be more convenient if a single value could be deleted from or added to the existing values. It would also be advantageous to be able to "add" data modules together. A new data module would be created by adding together the values in different data modules that are associated with the same variable name. This capability would be of assistance in accumulating product-wise data for a division and divisional data for a corporation.

CMS/1 is designed for use in the "batch" mode. That is, all statements required for the execution of a model or models are grouped together, and then the entire sequence of statements is processed by CMS/1 without further intervention by the user. This mode of operation could be changed by altering CMS/1 so that it would operate in an "interactive" mode. An interactive system would process one statement at a time as they are entered rather than processing a "batch" of statements. The user could monitor the results of a model and immediately alter the data and/or the logic to analyze new alternatives. The model could thereby become an integral part of the analysis and decision-making process.

APPENDIX A

CORPORATE MODELING SYSTEM

RELEASE 1

Developed By

Robert F. Zant

Clemson University

USER'S MANUAL

INTRODUCTION

The Corporate Modeling System (CMS/1) is a computer-based system designed for use in corporate planning and budgeting. The user specifies the logical structure of a planning or budgeting problem (i.e., a logic module), the data to be used in executing the model (i.e., a data module), and the report format for the presentation of results (i.e., a report module). The modules are translated into a computer executable form and stored on magnetic disks(see Figure A-1). The modules may then be combined into a complete model and the model executed (see Figure A-2).

The user communicates with CMS through three types of statements - control statements, logic and data specification statements, and report format specification statements. The statements required for the execution of a simple model are presented in Figure A-3. The results of the execution are shown in Figure A-4.

Language Characteristics

Character Set

Models are written in the CMS languages using a 88-character set composed of digits, alphabetic characters, and special characters.

There are ten digits. The digits are the decimal digits 0 through 9.

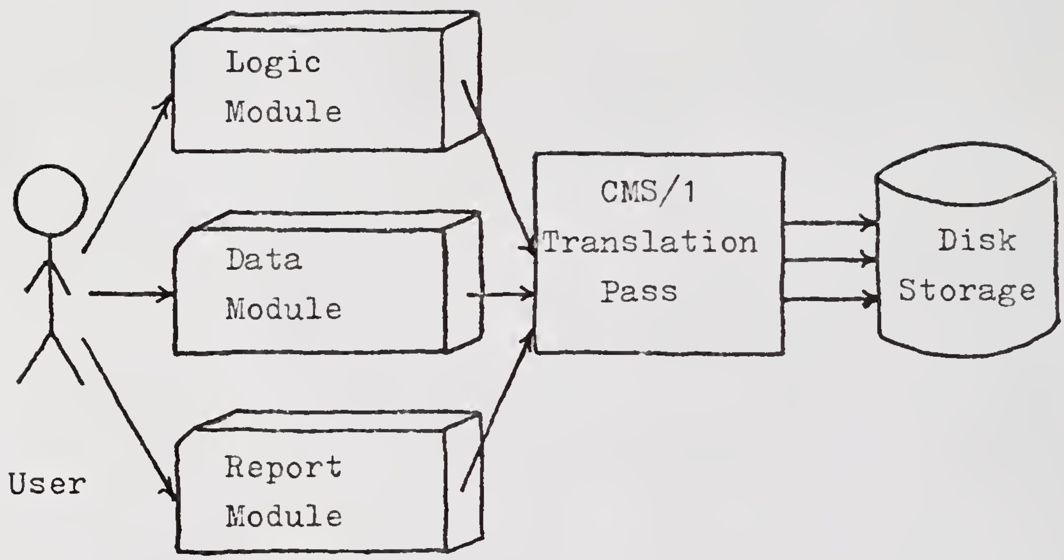


Figure A-1. Schematic of CMS/1 - translation pass.

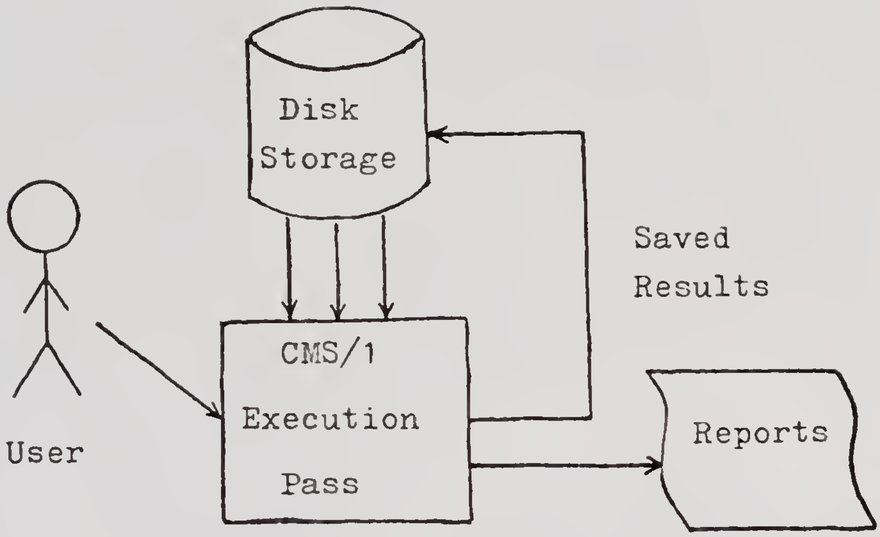


Figure A-2. Schematic of CMS/1 - execution pass.

```
*LOGIC  SAMPLE
 SALES ARE GROWTH(SALES, , GROWTH_RATE)
 COST_OF_GOODS_SOLD IS SALES TIMES COST_PERCENTAGE
 GROSS_MARGIN EQUALS SALES MINUS COST_OF_GOODS_SOLD
 OPERATING_PROFIT IS GROSS_MARGIN MINUS GEN_ADMIN_EXPENSES
1              MINUS DEPRECIATION
 INCOME_TAX EQUALS TAX_RATE TIMES OPERATING_PROFIT
 NET_INCOME EQUALS OPERATING_PROFIT MINUS INCOME_TAX
 CASH_FLOW EQUALS NET_INCOME PLUS DEPRECIATION
*DATA  PERIODS 1 TO 4
 SALES EQUAL  5000,
 GROWTH_RATE IS .15, .10, .05, .05
 COST_PERCENTAGE IS .75
 DEPRECIATION IS 400
 GEN_ADMIN_EXPENSES ARE 350
 TAX_RATE IS .50
*REPORT  INCOME
 LINE LENGTH  56
 MARGIN  0
 COLUMN SIZES 20, 4, (8)
 BEGIN NEW PAGE
 SKIP 2 LINES
 COLUMN HEADINGS "ACCOUNT" "UNIT" "   1972" "   1973"
1                  "   1974"  "   1975"
 DOUBLE SPACE
 ITEM  , "(K$)" , SALES
 ITEM  , "(K$)" , COST_OF_GOODS_SOLD
 ITEM  , "(K$)" , GROSS_MARGIN
 ITEM  , "(K$)" , GEN_ADMIN_EXPENSES
 ITEM  , "(K$)" , DEPRECIATION
 ITEM  , "(K$)" , OPERATING_PROFIT
 ITEM  , "(K$)" , INCOME_TAX
 ITEM  , "(K$)" , NET_INCOME
 SKIP 1 LINE
 ITEM  , "(K$)" , CASH_FLOW
*EXECUTE LOGIC SAMPLE  REPORT INCOME
```

Figure A-3. Example model.

ACCOUNT	UNIT	1972	1973	1974	1975
SALES	(K$)	5000	5500	5775	6064
COST OF GOODS SOLD	(K$)	3750	4125	4331	4548
GROSS MARGIN	(K$)	1250	1375	1444	1516
GEN ADMIN EXPENSES	(K$)	350	350	350	350
DEPRECIATION	(K$)	400	400	400	400
OPERATING PROFIT	(K$)	500	625	694	766
INCOME TAX	(K$)	250	313	347	383
NET INCOME	(K$)	250	313	347	383
CASH FLOW	(K$)	650	713	747	783

Figure A-4. Results from example model.

There are 52 alphabetic characters. They are the 26 capital letters and the 26 lower case letters of the English alphabet (lower case letters are automatically converted to capitals except in a few cases in which strings of characters are created solely for the purpose of printing them on reports).

There are 26 special characters. They are given in Table A-1.

The term "alphameric character" refers to an alphabetic character or a digit, but not a special character.

Statements

The exact forms and the meanings of the different types of statements are described in the sections on the individual languages. All statements are contained on lines that are seventy-two characters in length. If a statement cannot be entered as a single line, all lines after the first line must contain a one ("1") in the first position.

Lines with the character "C" in the first position are considered to be comment statements. Comments do not affect the execution of surrounding lines.

TABLE A-1

Special Characters in the CMS/1 Languages

<table>
<tr><th>Name</th><th>Character</th></tr>
<tr><td>Apostrophe or single quote mark</td><td>'</td></tr>
<tr><td>Blank</td><td></td></tr>
<tr><td>Break Character*</td><td>_</td></tr>
<tr><td>Cent Sign</td><td>¢</td></tr>
<tr><td>Colon</td><td>:</td></tr>
<tr><td>Comma</td><td>,</td></tr>
<tr><td>Commercial "at" Symbol</td><td>@</td></tr>
<tr><td>Decimal Point</td><td>.</td></tr>
<tr><td>Divide Symbol</td><td>/</td></tr>
<tr><td>Dollar Sign</td><td>$</td></tr>
<tr><td>Equal Sign</td><td>=</td></tr>
<tr><td>Exclamation Mark</td><td>!</td></tr>
<tr><td>Left Parenthesis</td><td>(</td></tr>
<tr><td>Minus Sign</td><td>-</td></tr>
<tr><td>Multiply Symbol</td><td>*</td></tr>
<tr><td>Number Sign</td><td>#</td></tr>
<tr><td>Percent Symbol</td><td>%</td></tr>
<tr><td>Plus Sign</td><td>+</td></tr>
<tr><td>Question Mark</td><td>?</td></tr>
<tr><td>Quote Marks</td><td>"</td></tr>
<tr><td>Right Parenthesis</td><td>)</td></tr>
<tr><td>"NOT" Symbol</td><td>¬</td></tr>
<tr><td>"AND" Symbol</td><td>&</td></tr>
<tr><td>"OR" Symbol</td><td>|</td></tr>
<tr><td>"GREATER THAN" Symbol</td><td>></td></tr>
<tr><td>"LESS THAN" Symbol</td><td><</td></tr>
</table>

*Same as the underline character on typewriters.

SYSTEM CONTROL LANGUAGE

The system control language is composed of seven control statements. Each control statement begins with an asterisk in the <u>first position</u> followed by a control word. The control word may be followed by a combination of parameter keywords and parameter values.

Although the system control language does not distinguish between upper and lower case letters, the following explanation of the control statements indicates system defined words by upper case letters, and indicates user-originated words and variables by lower case letters. In the following section the form of each control statement will be given along with an explanation of the meanings of the parameter sets which may appear on each statement.

<u>Statements</u>

*LOGIC logic_module_name

The LOGIC statement specifies that the following statements (up to the next control statement) define the logic module for a model. If a logic module name is specified, the module will be saved in an executable form. An unnamed module is available only until another LOGIC statement which does not contain a name is encountered or until the end of the computer run.

*DATA data_module_name, PERIODS x TO y

The DATA statement specifies that the following statements (up to the next control statement) define data values to be used in the execution of a model. Each variable in the data module will be defined for 1+y-x periods. If the PERIODS option is not specified, then the default values of 1 TO 5 are assumed. If a data_module_name is specified, the data module will be saved in an executable form. An unnamed data module is available only until another DATA statement is encountered which does not contain a data module name or until the end of the computer run.

*REPORT report_name

The REPORT statement specifies that the following statements (up to the next control statement) define the format for printing the results from the execution of a model. If a report name is specified, the format specification will be saved in an executable form. An unnamed report is available only until another REPORT statement is encountered which does not contain a report name or until the end of the computer run.

*EXECUTE LOGIC logic_module_name, DATA
1 data_module_name$_1$, data_module_name$_2$, REPORT
1 report_name$_1$, report_name$_2$, PERIODS w TO z,
1 ITERATIONS n, INITIAL RANDOM NUMBER IS m,
1 HEADINGS "heading one", "heading two", SAVE
1 data_module_name

The EXECUTE statement specifies that the named logic module is to be executed using the data modules named. The results of the execution are printed according to the format specified in the named reports. If the LOGIC and/or REPORT options are not specified, the last unnamed logic module and/or report are used. The last unnamed data module is always used. The model is executed over the periods w to z. If the PERIODS option is not specified, the number of periods specified in the first data module is used.

When the HEADINGS option is specified, the first heading is centered at the top of each page of output and the second heading is printed against the right margin at the top of each page of output. Lower case letters appearing in the body of a heading are not translated into capitals. The body of the headings must be preceded and succeeded by quote marks.

The ITERATIONS and INITIAL RANDOM NUMBER options may be specified when Monte Carlo sampling is utilized. The default values are 1 and 65549, respectively. The ITERATIONS option is also used to specify the number of executions to be performed in a sensitivity analysis.

The SAVE option is used to specify a name for the data module which is created to contain the values of the variables appearing in SAVE statements in the logic module executed.

*CONTINUE

The CONTINUE statement may be used to denote the ending of a logic module, data module, or report module. If a CONTINUE or another control statement immediately follows a LOGIC [DATA, REPORT] statement, then the logic module [data module, report module] will be null.

```
*PRINT  ALL

*PRINT  NONE

*PRINT  LOGIC, DATA, REPORT
```

The PRINT statement governs the printing of statements read by CMS/1. If no PRINT statements are used, all statements read by CMS/1 will be printed. All control statements read will always be printed. The NONE option causes the cessation of the printing of logic, data, and report specification statements. The ALL, LOGIC, DATA, and REPORT options are used to restore the printing of statements read by CMS/1.

```
*DESTROY  LOGIC logic_module_name, DATA
1    data_module_name, REPORT report_name
```

The DESTROY statement is used to erase a named logic, data, or report which was previously created but no longer needed.

LOGIC AND DATA SPECIFICATION LANGUAGE

Logic and data modules are composed of statements in the logic and data specification language. There are seven types of statements in the language. Two of these types may be used to create data modules. All seven types may be used in creating logic modules.

Statements are composed of identifiers, operators, and delimiters.

Identifiers

An identifier is a single alphabetic character or a continuous string of alphameric and certain special characters. The first character in the string must not be a digit. The beginning and the end of an identifier are denoted respectively by the occurrence of preceding and succeeding delimiters and/or operators.

The special characters which may be used in an identifier are: apostrophe ('), break character (_), cent sign (¢), commercial "at" symbol (@), dollar sign ($), exclamation mark (!), percent symbol (%), and question mark (?).

Identifiers are used as names of variables, names of groups of variables, and names of statements. No identifier can exceed 32 characters in length. If an identifier longer than 32 characters is specified, it is contracted by concatenating the first 28 and the last 4 characters.

Certain identifiers are language Keywords. Keywords have specific meanings and cannot be used as names or labels. The Keywords and their meanings are discussed below in the section on statements.

Operators

Operators are single characters or strings of characters which specify arithmetic, logical, comparison, or assignment operations. The operators and the actions they connote are shown in Table A-2.

A special type of operators called function operators cause a predefined sequence of actions (usually arithmetic) to occur. The available functions and their descriptions are given in the section entitled "Functions."

Constants

Constants are strings of numeric characters which represent decimal numbers. The string may therefore contain one decimal point. Examples of constants are:

1.23
100
6345

A constant cannot contain blanks or commas.

All values in CMS/1 are maintained with a precision of 6 digits. All values must be between approximately 10^{-78} and 10^{+75}.

Delimiters

Delimiters are characters which denote the limits of

TABLE A-2

Summary of Operators in the Logic and Data Specification Language

Type	Action	Operator	Alternative Forms
Arithmetic:	addition	+	plus
	subtraction	-	minus
	multiplication	*	times
	division	/	divided by
	exponentiation	**	exponentiated by
	summation		sum of
Logical:	negation	¬	not
	and	&	and
	or	|	or
Comparison:	(not) greater than	(¬)>	IS (ARE) (NOT) GREATER THAN
	(not) less than	(¬)<	IS (ARE) (NOT) LESS THAN
	(not) equal	(¬)=	IS (ARE) (NOT) EQUAL TO
			EQUALS
	greater than or equal to	>=	
	less than or equal to	<=	
Assignment:	Assigns value	=	IS (ARE)
			EQUALS
			(IS) (ARE) EQUAL

Note: Blanks may not occur in the composite symbols, "**", "¬>", "¬<", "¬=", ">=", and "<=".

identifiers, of constants and of combinations of identifiers, constants,and operators. Characters which may be used as delimiters are the blank, comma, quote marks, number sign, colon, and left and right parentheses.

A blank, when appearing in the first position of an input line, denotes the beginning of a new statement (see the following section on Statements). One or more blanks may also be used to surround identifiers, operators, constants, or other delimiters. However, identifiers, composite operators, and constants cannot contain blanks.

NOTE: The first position of an input line must contain either a blank, a "1," or a "C."

Commas are used primarily to separate variable names in lists of variable names and to separate constants in lists of constants. Usually, a comma could be replaced by a blank.

The number sign is used to separate a variable name and an associated subscript. For example, if "SALES" represents the dollar value of sales for this period, "SALES#-1" would represent the dollar value of sales for the pervious period. A value of zero is assumed for a subscripted variable when the subscript references a period outside the range of the model.

Pairs of left and right parentheses enclose variable names, operators, and constants, creating expressions whose values are to be determined. An unmatched right parenthesis placed after the first identifier on a new line causes the identifier to be interpreted as a statement label.

A colon is used to separate a group name from the name of a variable that is contained in the group (see the explanation of the GROUP statement).

In the logic specification language quote marks are used in the SAVE statement in constructing a name for the saved variable (see the explanation of the SAVE statement).

Expressions

An expression is a representation of a value. A constant, a variable, and a combination of constants, variables, operators, and parenthesis are expressions. For example, the following are expressions:

```
43.2
SALES
A + B
(A + B) * C
```

A logical expression is an expression which assumes a "value" of "True" or "False." Example of logical expressions are:

```
SALES   1000
(X = 3) & (Y IS 5)
```

Statements

The statement is the basic executable unit. Statements compute values and assign them to variables, affect the sequence in which statements are executed, create associations (groups) of variables, and do nothing. *Only Assignment* and *GROUP* statements may be used in specifying a *data module*.

Assignment Statement

The assignment statement consists of a statement name (optional), the Keyword "SET" (optional), a variable name, an assignment operator, and an expression. Examples of assignment statements are:

```
W IS 1
SET X = A + B * Y
START) SET WAGES EQUAL TO HOURS TIMES PAY_RATE
```

Control Statements

There are three types of control statements - the GO TO, IF, and STOP statements. The GO TO statement is composed of a statement name (optional) and the Keywords "GO TO" (or "JUMP TO") followed by the name of another statement. The execution of a GO TO statement causes a sequential execution of statements to begin with the named statement. Examples of GO TO statementsare:

```
JUMP TO FINISH
GO TO START
GO_1) GO TO LOOP
```

The second type of control statement, the IF statement, is composed of a statement name (optional), the Keyword "IF," a logical expression, the Keyword "THEN," an unnamed statement, the Keyword "ELSE," and an unnamed statement. The components must appear in that order. The Keyword "ELSE" along with the last statement is optional.

The execution of an IF statement causes the logical expression to be evaluated. If the expression is TRUE, the statement following the Keyword "THEN" is executed. If the expression is FALSE, the statement following the Keyword "ELSE" is executed. Examples of IF statements are:

```
 IF SALES ARE 1000 THEN GO TO LOOP
1    ELSE C_G_S = .8 * SALES
 TEST) IF (X IS 3) & (Y = 5) THEN X = 1.5
```

The statement following a "THEN" or an "ELSE" cannot be an IF or a GROUP statement.

The third type of control statement, the STOP statement, is composed of a statement name (optional) and the Keyword "STOP." The encountering of this statement causes the execution of the model to cease. An example of the STOP statement is:

```
THE_END) STOP
```

<u>Null Statement</u>

A null statement is nonexecutable. It consists only of a statement name. Examples of null statements are:

```
LOOP)
BEGIN)
```

<u>GROUP Statement</u>

The GROUP statement associates variables with group names. The use of a group name in an expression or in an assignment statement will cause each variable associated with the named group to be referenced.

For example, the two statements

```
GROUP GROSS_MARGIN, EXPENSES,
1    NET_INCOME_FROM_OPERATIONS UNDER DIVISION_1,
1    DIVISION_2, DIVISION_3, DIVISION_4
NET_INCOME_FROM_OPERATIONS EQUALS GROSS_MARGIN MINUS
1    EXPENSES
```

would compute the value of four variables, i.e., the net incomes for the four divisions.

The value of a single variable defined by a GROUP statement may be referenced by specifying the group name and the variable name separated by a colon. For example,

```
DIVISION_1:GROSS_MARGIN
DIVISION_2:GROSS_MARGIN
DIVISION_4:EXPENSES
```

would reference individual values. In this manner the twelve values could be individually referenced.

GROUP statements, if used, *must appear before* any other type of statement.

The use of group and unqualified element names in assignment statements is interpreted as follows:

Group name = f(group name)

> The statement is expanded into statements for each element in the group. The groups must have the same number of element names. The element names are paired in order of their occurrence in the GROUP statement.

Group name = f(variable name)

> The value of the expression is assigned to all members of the group.

Group name = f(element name)

The statement is expanded for each element in the group. The element named is summed over all occurrences.

Variable name = f(group name)

The sum of the values of the elements in the group is used in the expression.

Variable name = f(element name)

The sum of the element's value over all groups is used in the expression.

Element name = f(group name)

The statement is expanded into statements for each occurrence of the element name. The sum of the values of the elements in the group is used in the expression.

Element name = f(variable name)

The value of the variable is assigned to each occurrence of the element name.

Element name = f(element name)

If both elements belong to the same groups, then the statement is expanded into statements for each group.

If the elements belong to different groups, then they must belong to the same number of groups and the statements are expanded into statements for pairs (or vectors) of groups with the pairing being determined by the order in which the groups were listed in the defining GROUP statements.

SAVE Statement

The SAVE statement has the following form:

SAVE variable_name_1 AS variable_name_2, variable_name_3
1 AS variable_name_4, ...

The final values associated with the first variable name are associated with the second variable name and are written on a disk file. If the Keyword "AS" and the second variable names do not appear, the values written out will be associated with the first (i.e., the original) variable name.

A variable name which follows the Keyword "AS" may have the following form.

$$\text{VARIABLE\_NAME}_1\text{"VARIABLE\_NAME}_2$$

The values will be saved under a name created by appending to the first variable name the value of the variable referenced after the quote marks.

For example, if a model is being solved for a particular product (say product number 546) and the net income computed by the model is to be saved, then

```
PRODUCT_NUMBER IS 546
SAVE  NET_INCOME AS NET_INCOME_"PRODUCT_NUMBER
```

would save the values associated with "NET_INCOME" under the name "NET_INCOME_546".

If the "AS" clause is not utilized in the SAVE statement, then a variable may be appended to the original variable name. For example,

```
PRODUCT NUMBER IS 546
SAVE  NET_INCOME"PRODUCT_NUMBER
```

would save the values associated with "NET_INCOME" under the name "NET_INCOME546".

If a group name is used instead of the first variable name, then the second name is considered to be a group name. The values of all elements in the group are saved and are associated with the original variable names and the new group name.

If an element name is used instead of the first variable name, it must be qualified.

The values are saved in a data module with the name specified in the EXECUTE statement. If no name is specified, a data module name is generated from the name of the logic module and the date.

When a deterministic model is executed, the last values associated with the variables are saved. If the model is executed with Monte Carlo sampling, then the values saved are the average values for the variable. If the model is executed in a sensitivity analysis, the last values are saved.

Functions

Functions are predefined sequences of actions which are invoked by the use of a function name in an expression. An example of the use of the function "NORMAL" is:

X = Y * NORMAL (a,b)

The execution of the preceding statement would replace the value of X with the value of Y multiplied by a normally distributed random variate.

In the following explanations of the supported functions, the lower case letters "a", "b", and "c" represent expressions.

NORMAL (a,b)

A value is drawn from a normally distributed population. Ninety-five percent of the population lies between the values of a and b.

UNIFORM (a,b)

A value is drawn from a uniformally distributed population. The population lies between the values a and b.

WEIBULL (a,b,c)

A value is drawn from a population described by a Weibull distribution. Ninety-five percent of the values lie between the values of a and c. One percent of the values lie beyond the value of c. The value of b is the most likely value to occur (i.e., the mode).

DISTRIBUTED (a_1,b_1,a_2,b_2, ...)

A value is drawn from a population which contains the values a_1,a_2,a_3, ... The probability that a_1 will be selected is b_1; the probability that a_2 will be selected is b_2; etc. NOTE: The b's must sum to one.

LINEAR (a,b,c)

A value is computed which is equal to the minimum of b and a+(n-1)c where n is the period number (i.e., 1,2,3,...). If c is omitted [i.e., LINEAR (a,b,)], then c is computed according to the formula (b-a)/m-1) where m is the total number of time periods.

GROWTH (a,b,c)

A value is computed which is equal to the minimum of b and $a(1+c)^{n-1}$ where n is the period number (i.e., 1, 2, 3, ...). Thus, the value will begin at a and grow by c percent a year until the value of b is attained. If c is omitted [i.e., GROWTH (a,b,)], then c is computed such that a constant percentage growth from a to b is realized.

DISCOUNT (a,b)

The present value of the cash flow, a, is calculated using the interest rate, b.

DEPRECIATE (a,b,c)

An investment, b, is depreciated by one of three methods. The type of depreciation is governed by the value of a:

a = 1 straight line
a = 2 sum of the year's digits
a = 3 double declining balance

When a equals 1 or 2, c is the salvage value of the investment. When a equals 3, c is the number of periods over which the investment is depreciated.

INTRL_RATE (a,b)

The internal rate of return that equates the investment, a, with the cash flow, b, is calculated.

RATE (a,b,c)

That equivalent rate of return is calculated that equates the investment, a, with the cash flow, b, when the cash throw-offs earn the rate of return, c.

PAYBACK (a,b)

The payback period is calculated for the investment, a, with the cash flow, b.

ITERATE (a,b,c)

The ITERATE function is used to perform a sensitivity analysis. The first time the model is executed the value a is used. The value used is then incremented on each successive execution by c until a maximum value of b is attained.

REPORT FORMAT SPECIFICATION LANGUAGE

The format of a report is specified by the use of statements which govern the size of the printed page, segment the page into columns, control the spacing of lines, specify the content of lines, and define titles and footings to be printed on each page.

All statements begin with one or more blanks followed by a Keyword, followed by information concerning the action requested.

In the following section the form of each statement will be given along with an explanation of the meaning of the statement and any parameters which may appear in the statement.

Statements

PAGE LENGTH x

The PAGE statement sets the length of a page to be equal to x number of lines excluding the length of any footing. A page length of 60 lines is assumed.

MARGIN x

The MARGIN statement causes each line to begin x number of positions to the right of the physical beginning of the line. A margin of 2 positions is assumed.

LINE LENGTH x

The LINE LENGTH statement sets the maximum number of characters per line. A length of 130 is assumed.

BEGIN NEW PAGE

The BEGIN statement signifies the end of the logical content of a page. When this statement is encountered, a footing, if one was specified, is written at the bottom of the page and a new page is begun.

PRINT PERIODS x TO y

The PRINT PERIODS statement signifies that only values for periods x through y are to be printed. If x and y are not given, the number of periods over which the logic module is executed is assumed.

COLUMN SIZES x_1 x_2 x_3 ...

The COLUMN SIZES statement segments the page into columns of widths x_1, x_2, x_3, etc. The widths must be between 0 and 32768. If no column sizes are specified, the first column is assumed to have a width of 32 characters, the second column is assumed to have a width of 4 characters, and all succeeding columns are assumed to be 10 characters wide. To signify the repetition of an ending series of widths, enclose the series in parentheses.

COLUMN HEADINGS "a" "b" "c" ...

The COLUMN HEADINGS statement causes the indicated character strings (i.e., a, b, c, ...) to be printed in the center of the corresponding columns. To signify the repetition of an ending series of character strings, enclose the series in parentheses. No character string should be specified if the corresponding column width is zero.

SKIP x LINES

The SKIP statement causes x number of lines to be skipped before the printing of the next line.

SINGLE SPACE

The SINGLE SPACE statement causes the following printed lines to be single spaced.

DOUBLE SPACE

The DOUBLE SPACE statement causes the following printed lines to be double spaced.

TRIPLE SPACE

The TRIPLE SPACE statement causes the following printed lines to be triple spaced.

ITEM "a", "b", x, variable_name

The ITEM statement causes the character strings a and b to be printed in the first and second columns, respectively. The values of the named variable are printed in the remaining columns with each value having x decimal places (zero decimal places are assumed). The character strings a and b and the value x are optional. If the first character string does not appear (i.e., the Keyword "ITEM" is followed directly by a comma), the name of the variable is printed in the first column.

DISTRIBUTION

The DISTRIBUTION statement is used when a Monte Carlo simulation has been performed. It signifies that all following variables listed in ITEM statements are to have their values printed in the form of a relative frequency table.

LINE

The LINE statement signifies that any characters (beginning with position seven) following the Keyword "LINE" are to be printed exactly as they appear.

TITLE

The TITLE statement signifies that any characters(beginning with position eight) following the Keyword "TITLE" are to be printed at the top of each following page.

FOOT

The FOOT statement signifies that any characters (beginning with position seven) following the Keyword "FOOT" are to be printed at the bottom of each following page.

BIBLIOGRAPHY

1. "AUTOTAB User Guide," CAPEX Corporation, 1970.

2. Boulden, James B. and Edward S. Buffa. "Corporate Models: On-line, Real-time Systems," Harvard Business Review, Vol. 48 (July-August, 1970), pp. 65-80.

3. Brown, David E. "Stages in the Life Cycle of a Corporate Planning Model," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 92-116.

4. Buchman, Donald L. "An Application-Oriented Computer Language for Financial/Economic Simulation," a paper presented at the XIX Internation Meeting of the Institute of Management Sciences, Houston, Texas, April 5, 1972.

5. Dickson, G. W., J. J. Mauriel, and J. C. Anderson. "Computer Assisted Planning Models, A Functional Analysis," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 43-70.

6. Dinter, Heinz. "A Theory of Computer Based Management Control," Doctoral Dissertation, Department of Management and Business Law, University of Florida, 1970.

7. Dyment, John H. "Financial Planning with a Computer," Financial Executive (April, 1970), pp. 34-46.

8. "Financial Analysis System User's Guide," Information Service Department, General Electric, 1970.

9. Gershefski, George W. "Corporate Models - The State of the Art," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 26-42.

10. Gershefski, George W. The Development and Application of a Corporate Financial Model, Planning Executives Institute, Oxford, Ohio, 1968.

11. Gershefski, George W. "What's Happening in the World of Corporate Models?" The Bulletin of the Institute of Management Sciences (June, 1971), pp. 43-45.

12. Johansson, Hank. "Marketing and Financial Planning," The Bulletin of the Institute of Management Sciences (February, 1971), pp. 50-51.

13. Lande, Henry F. "Business Planning by Computer," Doctoral Dissertation, Western Reserve University, Cleveland, Ohio, 1965.

14. Lande, Henry F. How To Use the Computer in Business Planning, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1969.

15. McKenney, James L. "Guidelines for Simulation Model Development," Information Systems Science and Technology, 1967, pp. 169-173.

16. McKenney, James L. "The Roles of Simulation Models in Planning," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 600-613.

17. Morgan, James I. "Planning Simulator 1: An Aid To Business Planning," a paper presented at the 1970 Summer Computer Simulation Conference, Denver, Colorado.

18. Morgan, James I., R. M. Lawless, and E. C. Yehle. "The Dow Chemical Corporate Financial Planning Model," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 374-394.

19. Planning Systems Generator II, Program Number 5734-XT1, International Business Machines, Armonk, New York.

20. Wagner, W. H., L. T. Akutagawa, and L. J. Cuneo. "Telecommunications Earnings Estimation Model (TEEM): An Evaluation," Corporate Simulation Models, Albert N. Schrieber (ed.), Graduate School of Business Administration, University of Washington, Seattle, Washington, 1970, pp. 396-430.

21. Watson, H. J. "A Long-Range Stochastic Cash Budgeting Model," Doctoral Dissertation, Florida State University, Tallahassee, Florida, 1969.

BIOGRAPHICAL SKETCH

Robert Franklin Zant was born September 28, 1943, in Valdosta, Georgia. His childhood was spent in Georgia and in Florida where he graduated with Honors from Palatka Senior High School, Palatka, Florida, in June of 1961. He received an A.A. degree with High Honors in June, 1963, from St. Johns River Junior College, Palatka, Florida. Mr. Zant then attended the University of Florida where he majored in mathematics. In August, 1965, he received from the University a B.A. with Honors.

After graduation, Mr. Zant worked with The Boeing Company as a scientific programmer and with the University of Florida Computing Center as a systems programmer. In September, 1967, he began graduate studies in Economics and Business Administration at the University of Florida as a NDEA Title IV Fellow. He received an M.A. in management in March of 1970. Since August, 1970, Mr. Zant has been an Assistant Professor of Industrial Management at Clemson University.

Mr. Zant is married to the former Sue Ann Frazier of Palatka, Florida, and has two daughters. He is a member of Phi Rho Pi debating society, Phi Kappa Phi honorary scholastic fraternity, The Institute of Management Science, the American Institute of Decision Sciences, the American Accounting Association, and the South Carolina Academy of Sciences.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy

William V Wilmot

William V. Wilmot, Chairman
Professor of Management

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

R. H. Blodgett

R. H. Blodgett
Professor of Economics

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

E. L. Jackson

E. L. Jackson
Professor of Economics

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

W. E. Stone

W. E. Stone
Professor of Accounting

This dissertation was submitted to the Department of Management and Business Law in the College of Business Administration and to the Graduate Council, and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August, 1972

Dean, Graduate School