AN INITIAL CONCEPT STUDY FOR A PRODUCT MANAGEMENT DECISION SUPPORT SYSTEM (PMDSS) SUPPORTING EXECUTIVES IN A MANUFACTURING, MARKETING AND DISTRIBUTION COMPANY

by

Irma White

Project and Report submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Systems Engineering

APPROVED:

P. M. Ghare

P. M. Ghare, Chairman

P. Ghandforoush

P. Ghandforoush

L. K. Harmon

L. K. Harmon, Jr.

May, 1990

Blacksburg, Virginia

AN INITIAL CONCEPT STUDY FOR A PRODUCT MANAGEMENT DECISION SUPPORT SYSTEM (PMDSS) SUPPORTING EXECUTIVES IN A MANUFACTURING, MARKETING AND DISTRIBUTION COMPANY

by

Irma White

Committee Chairman: P. M. Ghare
Systems Engineering

(ABSTRACT)

This paper is the initial design concept for a Decision Support System (DSS) which will allow executives and their first echelon staff to answer questions relating to product management and will support the analysis of management issues relating to manufacturing, marketing and distrubution by an executive on his Personal Computer (PC).

This paper is intended to satisfy two objectives:

1) describe all functional requirements that a completed Product Management Decision Support System (PMDSS) should satisfy

2) describe the technical approach to satisfying these requirements to show its feasibility and, to give enough details to support prototype software development

ACKNOWLEDGMENTS

The author would like to take this opportunity to thank those who have encouraged her in the completion of this Project and Report.

A special thanks to the committee members who made this project a learning, rewarding and enjoyable process. Above all, I wish to thank my parents, my grandmother, my brothers and Debbie Udokwere whose continued support enabled me to dedicate myself completely to my work and goals.

Irma White

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION AND NATURE OF PROBLEM

1.1 Background and Illustrative Application

Executives of many manufacturing firms lack real-time integrated decision and executive support systems focusing on the company's strategic goals. The strategic goals of a manufacturing-based company are usually centered around product management, whether its targeting various marketing segments, researching and developing new products or distributing its products and manpower to yield the greatest profit. Thus the problem is generic in nature.

Alcott Pharmaceuticals (AP), a typical manufacturer, sells over 2,000 products to thousands of hospitals and consumers nationwide and abroad, and operates in 25 companies.[1] An illustrative application is used to provide specific examples, rather than generalized theory. The management hierarchy for corporate headquarters consists of; project managers reporting to program managers, program managers reporting to program directors, program directors reporting to group directors, group directors reporting to division vice presidents, etc. Program is defined here as the aggregation of products into manageable segments. At the other end of the hierarchy, the Chief Executive Officer (CEO) has created a new position, Chief Information Officer (CIO), parallel to the Chief Financial Office (CFO), recognizing data as a company asset and time as a critical success factor.[2] The CIO is tasked with converting, synchronizing and

---

[1]Ralph H. Sprague, Jr., and Eric D. Carlson, Building Effective Decision Support Systems. (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), pp. 202-203.

[2] Efraim Turban, Decision Support and Expert Systems: Management Support Systems. (Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990), p. 388.

integrating the company's data into usable information at all corporate levels in a timely manner.

A number of Decision Support and Management Information Systems have been developed to address the needs of project managers and analysts, at the corporate level. Program Managers (PM) and above, henceforth referred to as the executive team or the user, lack the tools to execute and manage their programs more effectively and efficiently.[3] The CEO has asked the CIO to find a means to resolve this problem. This study was initiated at the CIO's request.

1.2 Current System

Typically, manufacturing, marketing and distribution companies employ a reporting procedure that updates upper management with financial information on a monthly basis. This official report of key measures often arrives when information is already outdated or aged, and is too rigid and limited to allow for in-depth investigation, and is not complete, particularly regarding non-financial information. If a manager wants to perform a sensitivity analysis on a decision support model, a member of the Systems staff would be contacted and often it would take days or even weeks to get a response.

1.3 Need for PMDSS

The purpose of A Product Management Decision Support System (PMDSS) is to provide a management tool for the executive team, and to assist them in their decision-making process.

The following capabilities have been identified by the user:

a. It should support 3 major groups within the corporate structure.

---

[3] Ralph H. Sprague, Jr., and Eric D. Carlson, Building Effective Decision Support Systems. (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), p. 98.

1) Research and Development Programs (R&D)

2) Business Planning Programs (BPP)

3) Manpower Planning Programs (MPP)

b. It should support the executive team in dealing with program management and executive level issues. (i.e., allow the PM to examine both real and hypothetical scenarios)

c. It should address the following 5 basic types of questions:

1) "What-Is" - here the user is interested in examining Alcott's health and other industry and economic conditions. What is Alcott's bottom line, what is the state of the economy, how does Alcott's stock compare within the industry, are typical questions the executive team may ask. To answer these questions, PMDSS will extract key data from operational and external systems. After answering the "what-is" question, the executive may wish to investigate ways to improve Alcott's health.

2) "What-if" - the user wants to see the effect on the projection of changing specific parameters. For example, the user may wish to see the effect on projected net income by increasing projected sales by 10% each year.

3) "How-to" - the user wants to determine how to meet certain goals and/or constraints through modifying parameters. For example, the user desires a certain net income by next year and may want to determine the sales necessary to achieve this goal.

4) "What's-best" - the user wants to determine how to "best" meet certain goals and/or constraints through modifying specific

parameters. Here the user is willing for all his preferences, goals and constraints to be precisely formulated so that an optimization model may be used. The user does not have to input this information every time a question is asked. Instead, he modifies the inputs that should be changed from their default values. The "what's-best" question is a more restricted type of "how-to" question. For example, the user may want to determine sales as for the how-to question, while applying precise penalties for an advertisement budget.

5) "Why" - here the user wans to know why certain results were not achieved in the projection. To answer a "why" question, the PMDSS would have an expert system which contains the rules that a human expert would use to determine the cause of a projected result. For example, the system would have to know all of the potential causes for a drop in sales below company-wide goals. The ability to address "why" questions is the most difficult, and "why" questions are only answered as the result of answering the previous three types of questions.

d. A single unassisted user on his own PC should be able to use it. It should provide all capabilities interactively through user-friendly menus and input/output displays.

e. Typical projections should take from a few seconds to an hour of clock time, depending on the complexity and level of detail of the projection. The design should minimize the required execution time needed to

capture the important effects of and major interactions among the projection parameters.

f. The modules used in the PMDSS should incorporate the important functional parameters, goals and constraints used to manage and track the corporate business strategy.

1.4 Benefits of PMDSS

It is anticipated that the PMDSS will be operated by the user for two related purposes:

a. To directly answer questions which do not require the additional accuracy and detail of a production run made by the system staff.

b. To reduce the number of production runs needed to achieve a desired result.

The quick response time afforded by the PMDSS will encourage the analyst to ask more questions and allow him to see the effect to wider changes in constraints, etc., than he would normally see during the course of production runs. This capability will help train users to understand the business program strategy process and provide insight into how changes in input parameters can change projection results.

The benefits of the PMDSS fall into three broad categories:

- Increased information quality by allowing time for additional analysis of the effect of changing important parameters. The understanding of how results are affected by changes in parameters is useful in creating more robust plans.
- Improved analytical insight. The ability to see the effect of change in parameters in a matter of minutes will encourage the analysts to ask questions.

- Increased user productivity by supporting the faster achievement of the desired production results and by supporting quicker answers to user questions.

In addition, the PMDSS could be used as a platform for developing prototypes of models and user interfaces which would later be incorporated in other Management Support Systems.

Also, in the long term, it is likely that other Management Support Systems will utilize PC workstations for the user interface, with large data base and modeling components residing on a mainframe environment. In this case, transfer of interactive applications from the PMDSS to other Management Support Systems would be relatively easy, as they would be in the same hardware environment.

1.5 Description of the Proposed PMDSS

To make forecasts quickly, the PMDSS will be algorithms which are not as precise as in the mainframe based Management Support Systems. However, the PMDSS forecasts will capture the major effects pertaining to business strategy.

The PMDSS modules will include the dimensions most significant for it's management. Such dimensions include:

- product
- region
- aggregate business
- product category
- business segments

The PMDSS models will incorporate the key parameters, goals and constraints used to functionally manage the business, such as:

- budget constraints
- cost estimates
- scheduling constraints

The PMDSS will consist of five major PC-resident components and interfacing components on the VM and MVS mainframe computers. This is depicted in Figure 1.51. The major PC-resident components are:

1) Interface[4]

This will be a menu-driven current interface with access to all current and forecasted outputs. Default parameters will be displayed so that user input is limited to the changes from the default projection scenario. Eventually, this interface could be enhanced to a graphics interface e.g., Microsoft Windows,

IBM Presentation Manager). The user will be able to select subsets of user scenarios to modify and save the new scenarios. Users will then have the PMDSS execute the required modules. Then, projection results may be viewed and compared with other projections. All input data and output data may be compared across alternatives to show the cause and effect of changes in inputs from one alternative to another. The user interface could eventually support graphical displays as well as tabular ones, and provide for import and export of current data from external programs (e.g., LOTUS 1-2-3), if these options are desired.

---

[4] Ralph H. Sprague, Jr., and Eric D. Carlson, Building Effective Decision Support Systems. (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), p. 200.

2) Model Manager[5]

This software will interpret user input data and execution options from the User Interface and execute the appropriate mathematical models or procedures. The model manager will automate the model execution process, which will be entirely on the user's Personal Computer (PC)

---

5 Ibid., p. 260.

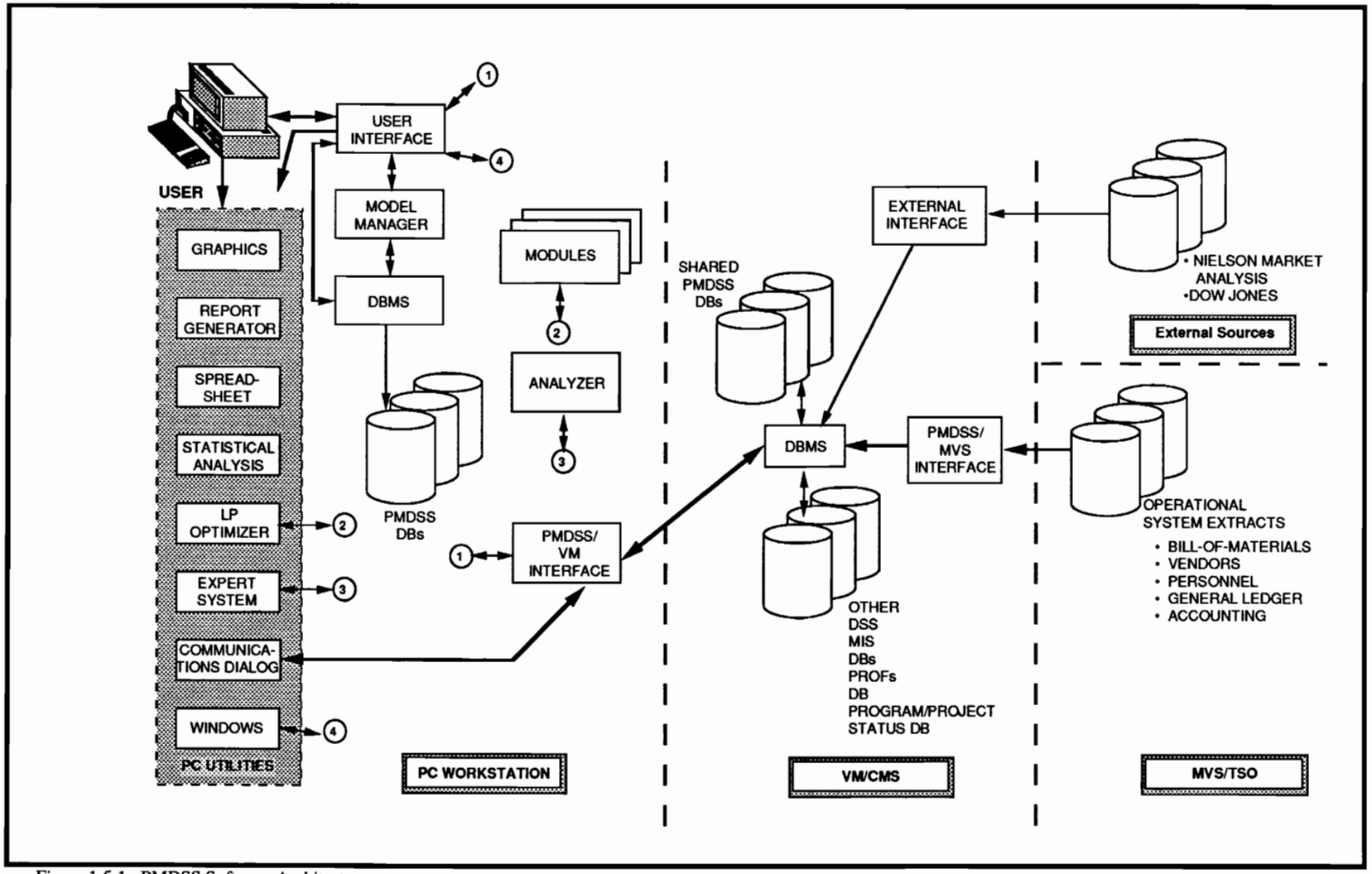


Figure 1.5-1. PMDSS Software Architecture
Source: Efraim Turban, Decision Support And Expert System; Management Support Systems
(Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990), p. 716.

workstation. Models may be executed immediately after all input modifications are made by the user.

3) Modules

These form a network of interconnected modules which incorporate the functional management structure of AP. Each module is oriented to performing a specific function at a particular level of detail. Data may be passed from one level to another to show the effect at one level on another level.

The modules will use several different mathematical techniques:

- risk analysis simulation to evaluate capital investments with uncertain outcomes.[6]
- goal optimization - the user will have the option of optimizing key decision variables to minimize deviations from the targeted budget.
- dynamic programming to maximize total return by optimally allocating resources to various programs in stages.[7]
- integer programming (capital budgeting) to aid user in project selection.[8]

---

[6] David Whitaker, OR on the Micro (John Wiley and Sons, Whichester, 1984), p. 129.

[7] Larry M. Austin and James R. Burns, Management Science; An Aid for Managerial Decision Making (Macmillan Publishing Company, New York, Collier Macmillan Publishers, London, 1984), p. 36.

[8] Ibid., p. 35.

Simulation will be available as a separate option, where the user wants to see he effects of changing parameters without reoptimizing. Simulation alone will be much faster in many cases.

The user will have access to the modules at the different major functional levels. Only the models necessary to answer specific questions need be executed. To execute on a PC, which as far less processing speed, memory and disk storage available than a mainframe, simplifications must be made, in some models. For example, projecting by quarter rather than month will reduce processing time by a factor of 3, for a given planning horizon. Simplifications have been made in all models from analogous production models. All of the modules are discussed in detail in the PMDSS Modules section in the Scenarios Chapter.

4) Projection Analyzer

This module will incorporate a rule-based expert system to allow an inexperienced user to apply expert knowledge in diagnosing the results of a projection, to determine why a certain result occurred or to find hidden problems in the projected results. This will be a later enhancement to the initial system.

5) Data Base Manager[9]

This module will manage the user's PC data base and also allow the export of data to other PC applications, such as LOTUS 1-2-3.

The PMDSS will have an interface with the VM and MVS environments. Baseline data will be provided by extracts of other management support systems and

---

[9] Ralph H. Sprague and Eric D. Carlson, Building Effective Decision Support Systems (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), p. 224.

external data on MVS environment. The VM computer will be used to pass data from one PC to others; it will act as the PMDSS "file server". The user will be responsible for downloading baseline data from the VM mainframe to his PC for the Business Planning, Research and Development and Manpower Planning Modules. The other data needed at a PC workstation will be downloaded automatically on a daily basis.

The system architecture of the PMDSS is discussed further in "PMDSS Architecture" in the Scenarios Chapter.

The PMDSS is initially targeted for a PC/AT compatible machine with math coprocessor, DOS operating system, 640 Kilobyte (Kb) Random Access Memory (RAM) and 15 Megabytes (Mb) of available hard disk space. The PMDSS software which is not obtained from commercial vendors will be programmed in the C programming language. C will be used because it is the most widely used portable and flexible programming languages for the PC environment. Much important commercial PC software is written in C, and there is a wide variety of off-the-shelf took kits for expediting C software development. In addition, C is being supported as a cross-platform language under IBM's Systems Application Architecture (SAA) for PCs, departmental computers and mainframes.

In the future, use of faster CPUs, larger available memory, windowing capabilities, etc., will allow features to be added to the initial capabilities. The hardware and software environment of the PMDSS is discussed further in "Hardware/Software Environment" under scenarios.

1.6 CEO Summary

The purpose of PMDSS is to provide a management tool for the executive team, and to assist them in their decision-making process by:

a. Supporting 3 major groups within the corporate structure.

1) Research and Development Programs (R&D)

2) Business Planning Programs (BPP)

3) Manpower Planning Programs (MPP)

b. Supporting the executive team in dealing with program management and executive level issues. (i.e., allow the PM to examine both real and hypothetical scenarios)

c. Addressing the following 5 basic types of questions:

1) "What-Is" - here the user is interested in examining it's health and other industry and economic conditions.

2) "What-if" - the user wants to see the effect on the projection of changing specific parameters.

3) "How-to" - the user wants to determine how to meet certain goals and/or constraints through modifying parameters.

4) "What's-best" - the user wants to determine how to "best" meet certain goals and/or constraints through modifying specific parameters.

5) "Why" - here the user wans to know why certain results were not achieved in the projection.

d. Allowing a single unassisted user on his own PC to use it.

e. Allowing typical projections to take from a few seconds to an hour of clock time, depending on the complexity and level of detail of the projection.

f. Incorporating the important functional parameters, goals and constraints used to manage and track the corporate business strategy.

The benefits of the PMDSS fall into 5 broad categories:

- Increased information quality by allowing time for additional analysis of the effect of changing important parameters. The understanding of how results are affected by changes in parameters is useful in creating more robust plans.
- Improved analytical insight. The ability to see the effect of change in parameters in a matter of minutes will encourage the analysts to ask questions.
- Increased user productivity by supporting the faster achievement of the desired production results and by supporting quicker answers to user questions.
- Better alignment of company-wide goals.
- Consistent measures of performance.

The PMDSS modules will include the dimensions most significant for it's management. Such dimensions include:

- product
- region
- aggregate business
- product category
- business segments

The PMDSS models will incorporate the key parameters, goals and constraints used to functionally manage the business, such as:

- budget constraints
- cost estimates
- scheduling constraints

The PMDSS will consist of five major PC-resident components and interfacing components on the VM and MVS mainframe computers. The major PC-resident components are:

1) Interface

This will be a menu-driven current interface with access to all current and forecasted outputs. Default parameters will be displayed so that user input is limited to the changes from the default projection scenario.

2) Model Manager

This software will interpret user input data and execution options from the User Interface and execute the appropriate mathematical models or procedures. The model manager will automate the model execution process, which will be entirely on the user's Personal Computer (PC) workstation. Models may be executed immediately after all input modifications are made by the user.

3) Modules

These form a network of interconnected modules which incorporate the functional management structure of AP. Each module is oriented to performing a specific function at a particular level of detail. Data may be passed from one level to another to show the effect at one level on another level.

The modules will use several different mathematical techniques:

- risk analysis simulation to evaluate capital investments with uncertain outcomes.

- goal optimization - the user will have the option of optimizing key decision variables to minimize deviations from the targeted budget.
- dynamic programming to maximize total return by optimally allocating resources to various programs in stages.
- integer programming (capital budgeting) to aid user in project selection.

The user will have access to the modules at the different major functional levels. Only the models necessary to answer specific questions need be executed. To execute on a PC, which as far less processing speed, memory and disk storage available than a mainframe, simplifications must be made, in some models. For example, projecting by quarter rather than month will reduce processing time by a factor of 3, for a given planning horizon. Simplifications have been made in all models from analogous production models.

4) Projection Analyzer

This module will incorporate a rule-based expert system to allow an inexperienced user to apply expert knowledge in diagnosing the results of a projection, to determine why a certain result occurred or to find hidden problems in the projected results. This will be a later enhancement to the initial system.

5) Data Base Manager

This module will manage the user's PC data base and also allow the export of data to other PC applications, such as LOTUS 1-2-3.

The PMDSS will have an interface with the VM and MVS environments. Baseline data will be provided by extracts of other management support systems and external data on MVS environment. The VM computer will be used to pass data from one PC to others; it will act as the PMDSS "file server". The user will be responsible for downloading baseline data from the VM mainframe to his PC for the Business Planning, Research and Development and Manpower Planning Modules. The other data needed at a PC workstation will be downloaded automatically on a daily basis.

The PMDSS is initially targeted for a PC/AT compatible machine. The PMDSS software which is not obtained from commercial vendors will be programmed in the C programming language. C will be used because it is the most widely used portable and flexible programming languages for the PC environment.

PMDSS will be incrementally developed to provide stepwise increase in capabilities. The prototype approach will initially be used to simulate PMDSS functions and to provide a means for user review. The initial development schedule is based on use of four full-time equivalents where the average cost of each full-time equivalent is $5,000/month. The schedule is from the Date Development is Begun (DDB).

- DDB + 3 months
  - provide prototype User Interface for Review.
- DDB + 6 months
  - provide prototype Critical Factors Module
- DDB + 7 months
  - provide prototype Executive Support Module
- DDB + 8 months
  - provide prototype BPP Module

- DDB + 9 months
  - provide prototype R&D Module
- DDB + 10 months
  - provide prototype MPP Module
- DBB + 12 months
  - convert prototypes to production Modules

It is estimated that the features listed in this concept study, including future enhancements and upgrades would take about 3 years to implement after DDB. The time estimates given above assume that three analyst are available full-time to work on this task, with part-time support from several others. The level of effort would be about 4 Technical Person Months (TPMs) per month for each month of the effort.

CHAPTER 2

SYSTEM GOALS

Some of the overall system requirements are:

- center the focus on the executive team on factors judged to be critical to the success of the enterprise.
- reduce the amount of information bombarding the executive team.
- increase the relevance, timeliness, consistency in presentation, and continuing usefulness of the information that reaches the executive team.
- enhance understanding of the data which is presented
- facilitate communicating with others in groups & on a one-on-one basis.
- allow ad hoc investigation into presented information

The PMDSS will be designed and developed to satisfy the following types of requirements:

- Detailed Decision Support Functions
- Executive Team Support Functions
- Performance requirements
- Hardware/software requirements

These requirements are listed in the following sections.

2.1 Decision Support Functions

The PMDSS will:

a. Support the existing executive team user community and other analysts dealing with program management issues.

b. Address four types of user questions, as defined below. These types of questions are functionally distinct, and may be answered using different techniques.

(1) "What-If" question - for the purposes of this concept study, a what-if question is defined to be where the user wants the Decision Support System (DSS) to determine the effect of his profitability changing the values of specific input parameters. For example, the user may want to see the effect of completing his accounts receivables in 15 days or less on projected profit.

For the "what-if" question, the user makes all of the decisions about which input parameters to change and by how much. Those not changed are left at their automatic default values. The "what-if" analysis may be used to find the sensitivity of important projection outputs (e.g., total budget) to changes in input parameters.

(2) "How-To" question - for our purposes, how-to questions require that the DSS select values for specified input parameters in order to reach specified goals and/or meet specified constraints. For example, the user may want to determine what changes are needed in quantity sold. This would be done in order to meet annual dividend goals. A "how-to" question is a "what-if" question "in reverse". A "how-to" question may be implemented as an optimization problem where quantitative or qualitative

goals and constraints exist .

For the "how-to" questions the user specifies or accepts default quantitative or qualitative goals and/or constraints, and also determines the range over which the input parameters may be varied in order to meet these goals/constraints. The "how-to" analysis may be used to find the sensitivity of projection inputs and outputs to the changes in goals and constraints. Goals may be specified for a large class of parameters.

(3) "What's-Best" question - This type of question is a special case of the "how-to" question. Not only does the user specify parameters that the model may modify to meet specific goals and constraints, but he also can specify numerical priorities. Using these priorities the user has implicitly formulated an objective function which may be optimized using Goal Programming (GP) or other mathematical optimization techniques.[10]

The "what's-best" question is asked when the user wants to modify a large number of parameters in the **"best"** way, given that goals are never completely achieved. The what's best capability implemented in the PMDSS will be used to determine parameters that explicitly minimize user-weighted deviations from profit targets.

---

10 Efraim Turban, Decision Support and Expert Systems: Management Support Systems (Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990), p. 326.

(4) "Why" question - for our purposes, this question requires that the DSS produce reasons why certain results were not achieved in the projection. In other words, what changes could be made in parameters, goals or constraints to increase profit. The "why" question is usually asked when the user does not understand the causes of a specific projected result. The "why" question is the result of a user's initial analysis of a projection, and so would always follow a "what-if", "what's-best" or "how-to" question. "Why" questions should be addressed by rules which emulate the analysis process of a knowledgeable analyst. Explicitly addressing "why" questions in the DSS is the primary way in which expert knowledge will be incorporated in the DSS. Rules will utilize user expertise.

c. Provide the user the ability to interactively update and view the key assumptions used for a particular scenario being modeled. Default assumptions will be available, so the user need only modify the assumptions that are different from the baseline. Quantitative assumptions that are parameters in the projection will be available for user modification and viewing.

d. Support the analysis of users who must have working functional knowledge of business structure but who will not need to have extensive knowledge of computers. This will be done through:

(1) Hands-on training for users will support two objectives:

- To explain the basic concepts and assumptions used internally in the PMDSS to transform user parameters, goals and constraints (i.e., the user's scenario) into projected results.
- To show how to actually navigate through the user interface to make the changes needed to define a scenario which will answer the user's question.

e. Produce projections using monthly, yearly or quarterly time periods. Reports may be specified for current time periods or any other available time period. The projection models should not be designed to have an inherent limit on the number of projected years. At the aggregate, total business level, up to 20 year projections should be supported. In most cases, projection scenarios will not be defined beyond 7 years.

f. Allow the user to isolate specific parameters for model modification or optimization . For example, the user may want to see the effect product categories optimizing for certain net profit only, keeping other product categories unchanged from the last scenario.

g. Allow the user to forecast at the specific level she is interested in (e.g., for a geographical region) without requiring that more aggregate level forecasts be made first.

2.2 Executive Team Support Functions

- assist in addressing the "What-Is" question
- provide an interface to IBM VM PROFS which allows users to electronically send and receive mail

- provide an interface to Dow Jones News and Nielson Market Analysis

2.3 Performance Requirements

The PMDSS will:

a. Provide interactive, user-friendly methods for defining user scenarios and for the reporting and analysis of the results of the personnel projection.

b. Provide default inputs for all user inputs and allow various scenarios. User input will only be necessary where changes in default values are needed.

c. Meet the following minimum performance requirements (all times are in terms of chronological time, not "machine time"). Times include the actual time to process user inputs, execute the projection, and display the results. They do not include the time the analyst needs to modify inputs or view outputs. All of the stated time requirements require the specified minimum PC configuration.

(1) The time required to answer a "what-if" or "what-is" question, as defined above, at a specific level of detail (e.g., aggregate business, a single product) for a specific year should be no greater than 1 minute. The time required to answer a question for n years should take no longer than n minutes (n is an integer from one to twenty).

(2) The time required to answer a "how-to" or "what's best" question, as defined above, at a specific level of detail for a specific year should be no greater than 5 minutes. The time

required to answer a question for n years should take no longer than 5n minutes.

(3) The time required to answer a specific "why" question, as defined above, should be no greater than 1 minute.

(4) The time required for any interactive operation in the user interface should be less than 10 seconds. Examples of interactive operations are:

(i) Selecting a menu or exiting a menu.

(ii) Accessing or storing user inputs on the PMDSS Data Base (DB).

(iii) Accessing projection results from the PMDSS DB.

Other, more complex operations, such as generating large multipage reports, may take longer. In addition, more complex user questions which may require the execution of several forecasts will require longer execution times.

Less detailed projections will require much less time than the maximum times listed above, assuming that most questions do not require all of the features embedded in the projection system.

d. Provide options that allow the user to forecast with acceptable tradeoffs between accuracy and speed of solution. In general, more accurate forecasting techniques require more computer time than less accurate ones, given that both are well designed.

e. Define "what-if" projections using baseline projections provided other Management Support Systems. The inputs to these baseline projections will be the default inputs to the what-if scenario. Hence, user input for

the definition of what-if scenarios will only be required specifically where the PMDSS scenario differs from the baseline case.

h. Allow easy comparison of user inputs or projected results of two different projection scenarios. All projected data types should be easily obtainable in a "delta" format, where the delta is the difference between the value for the 1st and 2nd projections. In addition, automated searches should be able to flag all of the deltas which exceed either a specific delta amount or a specific percentage delta. Deltas should be available for both projection inputs and outputs, as well as current inputs and outputs.

2.4 Hardware/Software Requirements

The PMDSS will:

a. Initially be developed for the IBM DOS environment (e.g., PC/AT, PS/2), because that is the only PC environment widely available to potential users. However, an approach must be presented for exploiting more powerful PC hardware and software environments as they are made available to the user community. These future capabilities would include:

(1) Graphical user interface (e.g., OS/2 Presentation Manager, MS Windows).

(2) More powerful Central Processing Units (CPUs) (e.g., Intel 80386, 486), larger and faster hard disks, and more available real memory.

(3) Simultaneous user sessions (e.g., as in OS/2) .

(4) Local Area Networks (LANs).

The addition of new PC platforms and software tools should not require major redesign of the initial PMDSS capabilities; i.e., all capabilities should be modular and as "upward-compatible" as possible.

b. Provide for initial implementation on the following minimum PC configuration:

(1) IBM PC/AT or 100% compatible.

(2) Disk Operating System (DOS) version 3.30 or higher.

(3) 640 Kb Random Access Memory (RAM) accessible to DOS and application programs (including the PMDSS).

(4) One 3 1/2 or 5 1/4 inch floppy disk drive and at least 15 Mb hard-disk space available for sole use of PMDSS data and program files.

(5) Ability to emulate 3270 terminal and link to VM computer, either through modem or dedicated line. Terminal emulation is typically done through use of specialized software (e.g., Attachmate EXTRA!). Mainframe to PC data transfer will only be done outside of the PMDSS session on the minimum PC configuration.

(6) 3270 emulators and other Terminate and Stay Resident (TSR) programs will not be active during PMDSS sessions. Only DOS and the PMDSS software will be active during that time.

(7) While a math coprocessor is not absolutely essential for use of the PMDSS, the required performance specifications will not be met without its use. In general, machines without math coprocessors will be limited to data input and output. A math

coprocessor typically increases the speed of floating-point computations by 5 times or more. The great majority of the execution time will be used to make floating-point calculations.

c. Support the functional analyst in a single-user workstation environment; i.e., all input to and operation of the what-if projection capability will be done independently by the PM on his own PC, though data may be passed from one user PC to another.

e. Allow data to be shared through:

(1) Uploading to/ downloading from the VM mainframe

(2) Via Local Area Networks (LANs) in the user's office

(3) Physical transfer of floppy disks

f. Use configuration management techniques to support data transfer[11] between several users using the PMDSS. Baseline and inputs from management support system, and forecast alternatives will all be labeled so that an audit trail can be maintained.

g. Use a structured, modular design so that additional models and other analytical tools may be added beyond those specified in this study. This will also allow for an efficient upgrade to more advanced hardware and operating systems.[12]

h. Provide for initial setup tasks in the IBM MVS and/or VM environments:

(1) Provide (monthly or daily, depending on application) loads of baseline historical and projection data to the shared PMDSS data

---

[11] Howard Eisner, Computer-Aided Systems Engineering (Prentice Hall Englewood Cliffs, New Jersey, 1988), p. 429.

[12] Audrey M. Weaver, Using the Structured Techniques: A Case Study (Yourdon Press, A Prentice-Hall Company, Englewood Cliffs, New Jersey 1987), pp. 114-118.

base from preprocessed on the MVS mainframe before it is loaded onto the VM computer for upload to the user's PC. The preprocessing will minimize the computational work required to load the baseline alternative onto the user's database.

i. Provide for automated transfer for data on the user's PC onto the shared VM computer environment, where it may be accessed by other users.

j. Provide for export of input or output data on the PMDSS DB to external PC applications such as LOTUS 1-2-3.

2.5 PMDSS Components

PMDSS consists of 6 components illustrated in Figure 2.5-1.

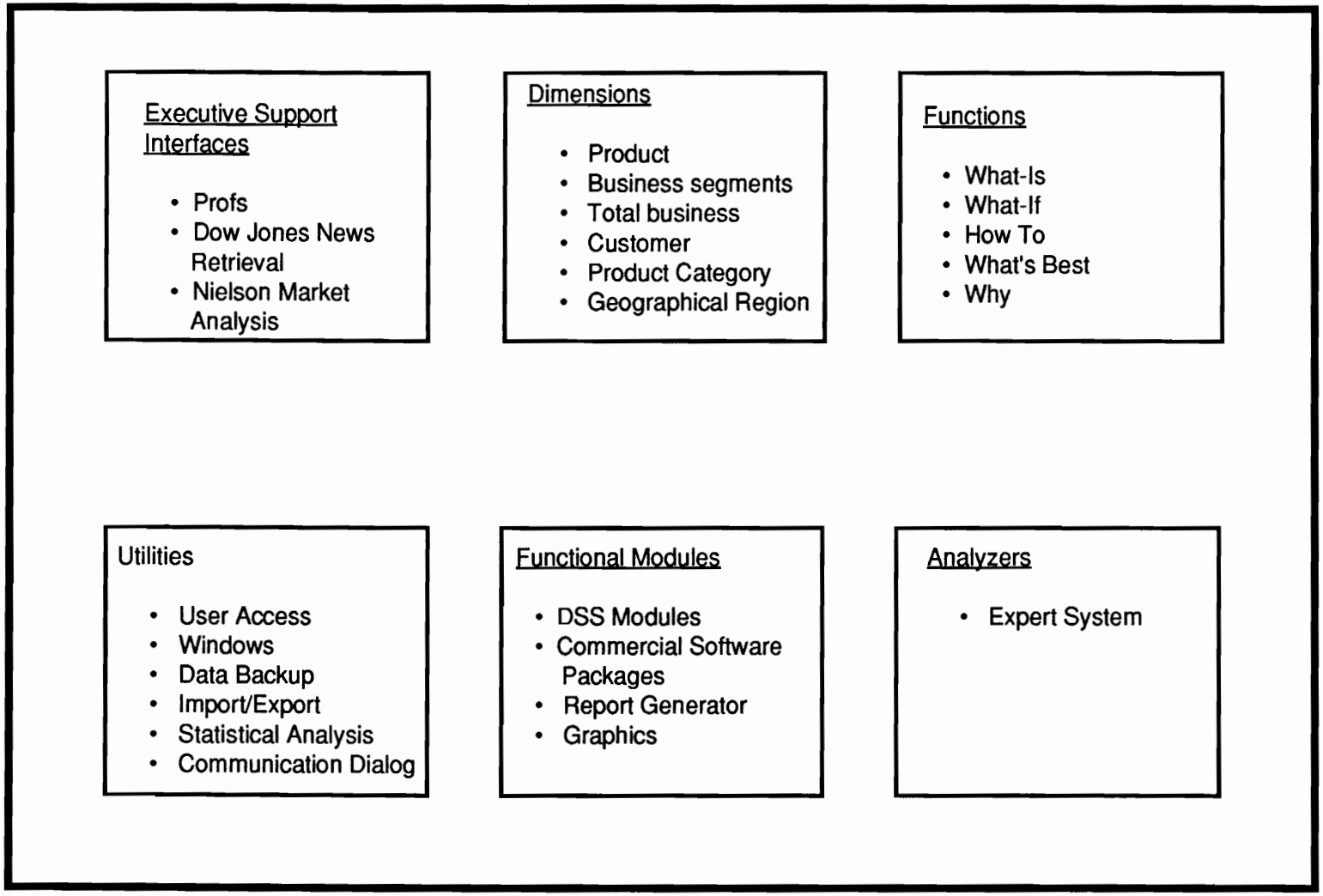


Figure 2.5-1. PMDSS Components
Source: Randall W. Jensen And Charles C. Tonies, Software Engineering (Prentice-Hall, Englewood Cliffs, New Jersey 1979), p. 156

CHAPTER 3

SCENARIOS

3.1 Hardware/Software Environment

This section describes the specific hardware and software alternatives for the development of the PMDSS and recommends choices, where it is currently appropriate.

3.1.1 Hardware Operating Systems

This section discusses:

- The primary hardware platform that will be used by the PMDSS.
- The operating system that will be used by the PMDSS.
- The procedural language used to program the functions of the PMDSS.
- The general role of the mainframe environment.

Primary Hardware Platform

The successful design and development of the PMDSS will require that key decisions be made on the target hardware and software used. The implementation of a "personal" DSS with a large set of analytical tools would be different if the target environment were, for example:

- an Intel 80386-based PC (e.g., IBM PS/2 Model 80) with the OS/2 operating system.
- an Intel 80286-based PC (e.g., IBM PC/AT) with the DOS operating system.
- an SUN 386 workstation with the UNIX operating system.

The most important decision to be made is the choice of the target operating system and of the minimum hardware configuration required. This decision should be based on the answers to these questions:

- What will be the mix of hardware platforms in use 5 years from now and what will be the operating system(s) used? This question is important because PMDSS should be targeted to a platform which will be available to the user in his office.
- What will the dominant class of PC hardware platforms and operating systems in the US commercial market 5 years from now? This question is important because the PMDSS should benefit from a rich variety of available commercial software which would be used, where possible, for some of its functions.

One design option is to build the PMDSS primarily in the PC environment. The PMDSS should be designed for a maximum useful life. It should not be designed specifically to be portable to the largest number of PCs currently available in corporate headquarters. On the other hand, the PMDSS should only require a type of hardware platform that will be procured as a part of the overall headquarters office automation process. In other words, unless there is a clear justification, special hardware should not be procured primarily for use by the PMDSS but should also support the other office automation functions.

Another design option is to target PMDSS to the CMS environment, either on the large VM mainframe (i.e., IBM 3087x series) or on the smaller departmental computers (i.e., IBM 937x series). This option carries all of the advantages of a multi-user environment:

a. Data may be more easily shared among users than in a "stand-along" PC environment.

b. An application could support inputs from multiple users.

c. Applications could be written in an environment similar to that of the current production systems.

d. Larger problems could be solved in this environment than in a stand-along PC environment.

and all of the disadvantages of a mainframe environment:

a. The response time of the PMDSS would be dependent on the overall system load. The delay caused in number-crunching applications, even for the time needed to move through the panels of the interface, could be significant. Large interactive multi-user systems tend to become saturated over time, no matter how much they are upgraded. There are always more computer intensive applications that users want, as long as the system resources are available.

Response time is important in two categories of processing: the interactive processing where the user is jumping from menu to menu or entering data, and the "batch" execution of projection models. Interactive sessions on mainframe environments, can be severely degraded by high system loads. A PC is usually a much better performer, because the user actually gets more computing power than he does when he is taking small "timeslices" of mainframe computer. The batch execution

performance of the MVS mainframe is usually much better than a PC's performance. However, there is the added overhead of time that a batch job has waiting to execute. In addition, there is the time needed to transmit the job from the VM to MVS and time needed to return the results. The additional overhead required to execute a batch job on the MVS computer is often 30 minutes or more, so jobs which would take less then 39 minutes to run on a PC would often be executed more quickly there. The additional operator controls required to run batch jobs also adds complexity to the mainframe option.

b. Given that there will typically be several PCs available in an office for use by a PMDSS user, the "perceived" reliability of the PMDSS on stand-alone PCs would likely be higher than that of the VM mainframe. While the reliability of an individual PC may be the same or less than a mainframe, the likelihood that all of the PCs in an office would be "down" is much lower than the likelihood that the mainframe would be down for that office.

Perceived reliability for a multi-user system must include the effect of the system being unavailable due to:

- scheduled system shutdowns
- dedicated processing for critical jobs
- scheduled and unscheduled maintenance

The "perceived" reliability is the likelihood that every PC in a collection of separate PCs would fail is the product of the failure probabilities of each PC. This is because PCs typically fail independently of each other. Thus, if the probability that one

PC would be down is .01, the probability that 3 PCs of that type would be down is (.01 * .01 * .01) or .000001. (i.e., mathematical probability[13]) that a system will be available to the user at the times when he needs to use it. Perceived reliability is really the system reliability of the hardware, including data lines and communication equipment.

System reliability is important for the PMDSS, because by definition the decision-maker needs to be provided quick answers to his questions. If one PC is down, the baseline data could be imported to another one via a backup disk.

Some mainframe systems are more fault-tolerant than others. For example, using multiple, interchangable, CPUs in the same environment greatly increases system reliability. But there are more individual components in a mainframe-based network (e.g., CPUs, front-end processors, controllers) that are subject to failure.

It is likely that PCs would not really be used in stand-alone mode. At headquarters, PCs are typically linked to the mainframe in 3270 emulation mode. Users can toggle between a PC session and a mainframe session almost instantaneously. In addition, it is anticipated that as PCs replace dumb terminals in an office, they will be configured into Local Area Networks (LANs), which can then link to the VM mainframe. Data can be shared among PCs on the LAN, or shared with other users via the VM mainframe. The PC design option must consider these communication/connectivity issues. Of course, PC should not depend on the mainframe being operational to perform the functions that reside in the PC session.

---

[13] Howard Eisner, Computer-Aided Systems Engineering (Prentice Hall Englewood Cliffs, New Jersey, 1988), pp. 108-109.

An individual PCs performance depends only upon applications it is processing, unless it also accesses shared resources (e.g., shared data on the LAN server, or required access to large files on the host mainframe). A PC is dedicated to the currently executing functions. These functions can use all of the CPU and disk resources on that PC, as opposed to sharing resources on a multi-user system.

Given the above points, it is preferable to develop the PMDSS for a PC environment if the PC platform made available by Alcott will have the capabilities needed to meet acceptable performance requirements and perform all of the required analytical functions.

The major PC performance characteristics are:[14]

- CPU processing speed - number of floating point operations per second (FLOPs)
- Read/Write time for the hard disk.
- Available Random Access Memory (RAM) for PMDSS use.
- Available storage space on the hard disk for PMDSS use.
- Additional features used to increase the utilization of the CPU such as a math coprocessor, cache memory.

The minimal PC configuration required by the PMDSS is listed in "Hardware/Software Requirements" in the System Goals Chapter.

---

[14] Ralph H. Sprague, Jr. and Hugh J. Watson, Decision Support Systems: Putting Theory Into Practice (Prentice Hall, Englewood Cliffs, New Jersey, 1986), p. 184.

The major PMDSS performance requirements is the maximum allowed execution time for each analytical tool. Based on this requirement, there is an inherent maximum model size allowed for each analytical tool (e.g., largest solvable LP model).

Operating System

There are three candidate PC operating systems for the PMDSS:

a. DOS - The original IBM PC operating system. Its major limitations are that:[15]

- It is designed for the 8 bit 8088 architecture; it does not fully utilize the newer 16 and 32 bit CPUs
- It does not allow memory of over 640 kb to be directly addressed.
- It is designed for a single user environment only.
- It does not support multiple applications running concurrently (multiprocessing). For example, the user could not answer his PROFS mail while the PMDSS model was executing in the "background". Concurrent processing, for the descriptions of multi-processing and multi-tasking, means that the multiple sessions or tasks are processed in the same period of time, but not that they are processed simultaneously. Single CPU PCs cannot execute multiple sessions or tasks at the same instant because they have only one processor. OS/2 and UNIX

---

[15] IBM. "Operating System/2 Extended Edition Briefing", Rockville, Maryland, November 1988, pp. 2-4 - 2-5.

implement "virtual" multiprocessing and multi-tasking by allocating time slices to the different processes or tasks.

- It does not support concurrent multiple tasks in a program (multi-tasking).

Its major advantages are that:[16]

- It is installed in over 90% of the existing PCs in the corporate headquarters environment.
- Much more commercial software is available in this environment, including packages which would be utilized as part of the PMDSS (e.g., LP optimizers, Data Base Management Systems (DBMSs)).
- It is a mature operating system.

It should be noted that there are add-on "DOS-Extenders", such as that produced by Phar Lap software. These allow more than 640 Kb to be addressed by applications using it. Other products, such as Microsoft Windows, incorporate multiprocessing within a DOS environment. But it should be noted that these products typically require a more powerful machine that the minimum PC configuration described in the previous section.

b. OS/2 - This is the most advanced IBM PC operating system, which is part of the IBM corporate Systems Application Architecture (SAA) strategy. The ultimate objective of SAA is to provide environments for

---

16 Ibid., pp. 2-1 - 2-3.

PC, minicomputer and mainframe systems that will allow applications to be developed for all three levels. The SAA strategy is in an early stage of development. The major advantages of OS/2 are listed below:[17]

(1) It allows memory over the 640 Kb limit to be addressed directly.

(2) It is designed to fully take advantage of the 16 bit CPU architecture (Intel 80286 processor). Eventually it will be expanded to utilize the 32 bit architecture of the Intel 80386 and 80486 chips.

(3) It allows multiprocessing.

(4) It allows multi-tasking.

(5) DOS applications may be run, but only in the active window.

(6) In its "Extended Edition", it includes, as part of the operating system:

- a Structured Query Language (SQL) based Data Base manager.
- a Query Manager, for making data base queries and developing reports.
- a Display Manager, for developing user friendly interfaces with panels and pop-up windows.

---

17 Ibid.

- a Communications Manager, which incorporates several communications methods (e.g., 3270 emulation) for communication with external systems. The more advanced communications features in OS/2 will ultimately support concurrent processing of a PC workstation and a mainframe under a PC application. This would allow the PC to call data from a mainframe data base while it is executing a projection model.

Its major disadvantages are:[18]

- It is early in its development cycle. Therefore, many features have not yet been released, and much less commercial software has been developed for it than for DOS.
- It requires faster PCs with more memory and disk space. It is oriented toward the upper-end 80386 based machines, not toward the 80286 based (e.g., PC/AT) level machines that are currently available to the corporate headquarter community.
- It is not designed to support multiple users.
- Some of the more advanced features, such as the Data Base Manager, will only be available on IBM PS/2s, or on other vendor's PCs with the Microchannel Bus architecture. The basic features of OS/2s, in its "Standard Edition" will be available for all PC/AT-compatible 80286 and 80386 based PCs, such as

---

[18] Ibid., pp. 2-4 - 2-5.

those manufactured by Zenith Corporation. OS/2 is also being sold directly by Microsoft, the co-developer of OS/2.

c. UNIX - This operating system was originally developed by AT&T. It was not developed for a particular hardware architecture, and has been implemented for various mainframes, minicomputers and PCs. The IBM variant of UNIX is AIX.

UNIX was developed to support multiple users on the same computer, and it also supports multiprocessing and multi-tasking. It also exploits the 32 bit CPU architecture, and allows almost unlimited addressing of memory (i.e., there is not 640 Kb limit. Unfortunately, few if any PCs in the corporate user community utilize UNIX. In the PC environment, UNIX has fewer available commercial application than DOS. UNIX is more popular for multi-user minicomputer systems.[19]

The Macintosh hardware/operating system was not considered for the target corporate PMDSS architecture because the installed base of the Macintosh machines in corporate headquarters is very small. Some apple machines allow execution of DOS applications; they may support the PMDSS, but this option has not yet been investigated.

Given the availability of commercial software, and the prevalence of PCs, DOS is the best choice for developing applications for PMDSS. However,

---

19 Ibid., p. 2-2.

the PMDSS should ultimately be developed to fully utilize the hardware which will be available 5 or more years from now.

Therefore, it is proposed that early development of the PMDSS capabilities be done in the DOS operating system, to give the user tools which he can use in his everyday analysis. As development progresses and more powerful hardware becomes available, new PMDSS capabilities will be written for the OS/2 operating system, and old ones will be converted from DOS OS/2. Software conversion should not be too onerous a task because the DOS and OS/2 operating systems have many similar features - in many ways DOS is a subset of OS/2[20] . It is not felt that UNIX will play a major role in the corporate community, although it is certainly possible.

3.1.1.1 Procedural Language

A general procedural language must be chosen for use in building all of the analytical tools in the PMDSS where more specialized commercial software (e.g., optimizers, DBMs) cannot be used, or where they are not as efficient[21] . It is possible that additional specialized languages will be chosen for use in building AI tools (e.g., PROLOG, LISP), if commercially available expert system shells do not provide all of the required AI capabilities.

---

[20] Ibid.

[21] Ralph H. Sprague, Jr. and Hugh J. Watson, Decision Support Systems: Putting Theory Into Practice (Prentice Hall, Englewood Cliffs, New Jersey, 1986), p. 177.

It should be noted that a significant effort is required to program applications under a commercial software package, and effort is required to integrate different commercial packages into a total set of decision support tools. Commercial packages generally cannot be treated as "black boxes" which can be immediately plugged into a system. They require significant programming effort to be utilized effectively[22]. There will be many functions that can most effectively be developed in a general procedural language.

The language chosen to build custom applications in the PMDSS should ideally be supported under DOS, OS/2 and any other possible future environment. It should be supported under the SAA architecture, so that applications written in it would ultimately be portable from the PC to the mainframe. In addition, the language chosen should have the greatest compatibility with commercial software which would be integrated into the PMDSS.

Given these conditions, the C language is the language of choice. A large percentage of the commercial software being written for PCs, whether under DOS, OS/2, UNIX or Macintosh, is being written in C. As a result, there is a large market for prepackaged C function libraries and other productivity-enhancing tools. C provides all of the essential features of a general high-level language. C is available under the MVS/XA operating system. In Addition, there is a superset of the C language, C++, which facilitates the development of object-oriented software.

---

22 Ibid., p. 179.

ALCOTT Pharmaceuticals currently has copies of the Borland Turbo C compiler, but Microsoft Professional C and Lattice C will be considered as well. All products support the standard set of C functions. However, Microsoft C is more accepted in the industry, and may be more compatible with external software (e.g., DBASE IV, Microsoft Windows). Lattice C has compatible compilers for the IBM, DOS, CMS and MVS environments.

3.1.1.2 Role of the Mainframe Environment

The VM and MVS mainframes will perform the task of preprocessing the input files from the Management Support Systems, and serve as the shared repository for data produced by all users of the PMDSS. All data for all models could not be stored on the average user's PC hard disk, given that the user may have multiple input versions and output alternatives for each model. Using the VM mainframe as a "file server" will give the user much greater data storage capability. The MVS computer will preprocess all of the baseline input data needed from the production Systems . This is discussed further in "VM Environment".

3.1.2 PC Software

The PMDSS will have several functions which will utilize commercial software. These functions are:

- User Interface
- Data Base Management
- PC-Mainframe Communication
- Optimization
- Expert System

- Report Generation

These software functions are described below. The design details of how these functions will be implemented in the PMDSS is covered in "PMDSS Architecture".

User Interface

The PMDSS user interface includes all functions needing user input and provides all user access to the output. For a typical session, the user will operate entirely within the PMDSS. Unless external software (e.g., a spreadsheet) is needed for a special analysis, the user will not need to use DOS commands. All menus and data displays for input/output will be consistent. The system will be designed so that a user must pass through a minimum number of menus to go from one part of the interface to another. A context-specific HELP facility will be available at any point in the system.

The PMDSS must provide its functions in a completely menu-driven format, using PF keys and single character commands (e.g., "s" for select)[23] . The prototype PC configuration will not support a "mouse"; interface, so a graphical "point and click" interface will not initially be developed. A graphical interface will not initially be developed in the DOS or the OS/2 environment, but it is not appropriate unless the user has a mouse on his/her PC. Microsoft Windows are the most popular DOS graphical interface environment. It extends DOS to include other features, such as multiple sessions. The OS/2 Presentation Manager, developed by Microsoft and IBM, is similar, but exploits the additional features of OS/2. Both products interface with C.

---

[23] Ibid., p. 98.

The prototype PMDSS user interface will be developed in the C programming language. C functions written by Alcott for other PC applications may be used as a template to efficiently construct this interface.

Data Base Management

This category includes the choice of a formal DBMS for maintenance of PMDSS system data (e.g., DBASE IV), and the choice of other storage techniques required for file management, where a formal DBMS is not appropriate. Intermediate solutions of an interactive projection model probably would not be stored on a DBMS, because the Input/Output (IO) time would significantly increase run times. A simpler filing system would be used instead (sequential or random access).

All applications in the PMDSS will need to access the PMDSS data base, so the choice of a compatible DBMS is critical for the successful implementation of all of the PMDSS tools. The DBMS should also provide the ability for the user to access data directly through the DBMS, through queries, or through setting up special reports.

DBASE IV, a product of Ashton-Tate, is a candidate for the PMDSS DBMS. It has the following features, which are requirements for use in the PMDSS:

a. It will execute reliably on the minimum PC configuration.

b. It allows records with up to 255 fields and 4 Kb.

c. It allows management of multiple data files, including relational updates and queries across different data files.

d. It allows the creation of user views of a DB for a particular user.

e It has a rich language for programming custom applications and can be compiled for faster execution. Customized menus and screens may be developed for an application. It incorporates a Structured Query Language (SQL) facility. The SQL used in DBASE IV is compatible with that used on the VM mainframe, so the VM user would not have to learn a different set of commands for the PC environment.

f. It interfaces with C.

g. While there is no built-in limit to the number of records, the largest file that could be processed effectively on the minimum PC configuration would be 10,000 records. This is greater than the length of any envisioned file.

The specific choice of DBMS will be made during the development of the prototype. Other candidates will be investigated (e.g., Nantucket Software's Clipper).

PC-Mainframe Communication

The initial PMDSS will use the 3270 emulation software available on the user's PC, when attached to VM controller or modem, to emulate an IBM 3270 series terminal. It can then access to both the MVS and VM computer environments. Files can be transferred from a PC session to a VM user session, and vice versa.

When it is feasible and desirable to upgrade the communication link between the user's PC and the VM mainframe, the current 3270 emulator may be replaced with a different communication software (e.g., the OS/2 Communications Manager).

Expert System

The PMDSS will need to utilize expert system techniques to answer "why" questions. There are two approaches to building such a capability[24] :

- Using an expert system shell. These are advanced "fourth-generation" software tools which allow the development of expert system rule bases and program the expert system.
- Using a general Artificial Intelligence language. These languages, such as LISP and PROLOG, allow many different types of "intelligent" systems to be programmed.

---

[24] Ibid., p. 149.

The expert system capability will not be developed as part of the initial PMDSS. More research will be made on which tools are appropriate as development of the PMDSS progresses.

Optimization

There are three separate functions performed by optimization software[25] :

a. Model formulation - a very compact language is used to represent the mathematical formulation of the optimization model, so that an analyst can quickly set up the mathematical structure of the problem to be optimized. Model formulation languages are typically based either on a table representation of the blocks of coefficients in the problem matrix, or on an explicit representation of the mathematical equations in the matrix. Model formulators exist for the generation of Linear Programs (LP), Goal Programs (GP) and Integer Programs (IP). Models may also be formulated through the use of spreadsheets.

b. Model Optimization - Model optimization is the process of reading in the model formulation in a standard representation such as MPS format, optimizing the model, and then outputting a standard report or set of data tables which may be read by an external program. Several optimizers allow input and output from specially formatted spreadsheets (e.g.,

---

25 Robert E. Markland and James R. Sweigart, Quantitative Methods: Applications to Managerial Decision Making (John Wiley and Sons, New York, 1987), pp. 251-261.

LOTUS 1-2-3), and several even include the capability to optimize small problems within the spreadsheet environment.

PC-based model optimizers can solve complex, realistic problems of significant size. The models that will be formulated as part of the PMDSS models will all be LPs, GP and IP models.

All of the LP models that are envisioned to be included in the PMDSS will be tractable using PC-based packages, at least where more powerful PCs are used (e.g., 80286 or 80386 CPUs with math coprocessors). LP models with the following dimensions can typically be optimized reliably by currently available PC software:

- Maximum of 1,000-2,000 rows
- Maximum of 2,000-4,000 columns
- Maximum of 10,000-20,000 non-zero coefficients

It should be noted that commercial PC Optimizers are making great leaps ahead as PC hardware becomes faster, so larger LPs will be solvable as PMDSS progresses. Use of 80386-based machines with more than 6740 Kb of addressable memory greatly increases the performance of LP solver, as they typically solve the LP all within the RAM memory.

Some current commercial optimizers can in theory solve larger LPs than those above (up to 8,000 rows and 16,000 columns), but LPs this large frequently have numerical problems and would not necessarily solve to optimality on the minimum PC configuration. Particularly dense

LP models (over 1% dense) are much more difficult to solve than sparser model with the same number of rows and columns.

$$\text{Density (in \%)} = \frac{(\text{\# of non-zero coefficients}) \times 100}{(\text{\# of rows}) \times (\text{\# of columns})}$$ [26]

c. Solution reporting - The standard reports generated by an optimizer generally are not very easy to read. Variables are labeled using internal names, the output includes information only of interest to an OR analyst, and the report is in a long sequential format that must be paged through. Some optimization packages include report generators which vastly improve the readability and utility of the solution. In addition, separate report generator packages may use the LP solution in the generation of output reports at the end of the projection process.

The following optimization software will be evaluated during PMDSS development, at a minimum:

- XPRESS-LP, a product of DASH Associates.
- CPlex, a product of CP Optimization, Inc.
- XA, a product of Sunset Software.

These packages all have a 80286 version and an 80386 version which allows use of more than the 640 Kb limit. The 80286 version and 80386 versions will be tested, but the version used on the minimum PC configuration will be for the 80286 CPU. The major evaluation criteria are:

---

[26] Thomas W. Knowles, Management Science, Building and Using Models, (Irwin, Homewood, Illinois, 1989), p. 219.

- Time required to solve a set of PMDSS test problems on both CPU types. This is important, because we want to have LP software which is upwardly compatible with faster PC platforms.
- Capability to link with C applications. All are available as callable function libraries.
- Accepts MPSX format input.
- Has a powerful model formulator and report generator for model development and debugging.

Report Generation

This function takes input data and formats reports for printed output that can easily be read and interpreted by the user. Report writers typically allow data to be arithmetically manipulated. Data from several sources can be combined into one report. Report writers usually include both tabular and graphical formats. A report generator must be able:[27]

- To format tabular reports to user specifications flexibly with headers, footnotes, different column formats, etc.
- To read data from the DBMS and file systems used in the PMDSS.
- To graph multiple data types and multiple what-if runs on the same chart.
- To produce presentation quality graphics.
- To support the range of printers used.

---

[27] Ralph H. Sprague, Jr. And Eric D. Carlson, Building Effective Decision Support Systems (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), p. 262.

- To be driven from control files. The user should not have to select graph options unless he wishes to deviate from the default case.

The generation of custom tabular and graphical reports will be an integral function of the completed PMDSS. This will be performed either through a stand-alone report generator or through use of the report generation capabilities of the DBMS chosen.

3.1.3 Supporting PC Software

There are other PC software tools that will be useful in analyzing PMDSS data on the user's PC. While they will not initially be an integrated part of the PMDSS, they will be able to access PMDSS data stored on the DBMS. The major categories of supporting software which may be used for analysis are:

a. Statistical Software

b. Spreadsheets

These categories are briefly discussed below.

Statistical Software

This software tasks formatted data files as input, and then performs specified statistical functions such as:[28]

a. Univariate statistics: mean, median, standard deviation, etc.

b. Time Series forecasting: exponential smoothing, seasonal techniques, Box-Jenskins methods.

c. Regression modeling.

---

[28] Howard Eisner, Computer-Aided Systems Engineering (Prentice Hall, Englewood Cliffs, New Jersey, 1988), p. 153.

d. Analysis of Variance (ANOVA) modeling.

e. Non-parametric statistical modeling.

Statistical software would be used by the analyst to postprocess projections of the PMDSS reduced models, or to process historical data or projections of the baseline production systems. Some statistical procedures can be "canned" as custom applications, so that the user need not know the statistical programming language. More sophisticated users may wish to write their own applications as needed. Statistical software generally incorporates report writing and graphics capabilities.

Time-series forecasts of projected sales forecasts for PMDSS will be done outside of the PC environment, as part of the mainframe interface from production systems. Therefore, there will be no need for imbedded statistical procedures in the PC-resident functions of the PMDSS.

Spreadsheet Software

The paradigm used for this interface is the accounting spreadsheet, with rows and columns labeled by the user for a particular data type. User-specified formulas define each particular data type. User-specified formulas define each row/column combination (i.e., cell) in terms of other cells. Spreadsheets may be customized by the user for particular applications, so that he need not spend time setting up the spreadsheet headings, data and formulas every time he uses the spreadsheet[29] . Many spreadsheets

---

[29] Steven C. Ross, Richard J. Perlesky And Lloyd D. Doney, Developing and Using Decision Support Applications (West Publishing Company, St. Paul, MN, 1988), p. 20.

now include the ability to define three-dimensional tables. Spreadsheets incorporate report generation and graphics for customized output of data.

The PMDSS will support output of data in LOTUS 1-2-3 and general ASCII text file format, so essentially all commercial PC spreadsheets in the DOS environment will be able to import PMDSS data.

3.2 PMDSS Architecture

The proposed system architecture for the PMDSS is depicted in Figure 1.5-1 . The proposed architecture follows from the assumptions that:

- The analytical functions of the PMDSS will reside on the individual PC workstations.
- The PC workstations will be linked to the VM mainframe environment, and hence can access the MVS mainframe environment.
- The VM environment will be used for shared PMDSS data storage.
- The MVS environment will be used to supply production data to the PC models from transaction and modeled-based production systems.

The PMDSS will execute independently of other production systems. The PMDSS will require a preload of the information derived from the production systems "baseline" runs.

In general, each PMDSS alternative is defined in terms of modifications to the input parameters (i.e., the scenario being projected) of a specific production system alternative.

The actual PMDSS projection is performed in the appropriate PM Module. Baseline data is input via the VM/MVS interface. This interface will allow data to be loaded through use of a simultaneous session on the VM mainframe (i.e., interactive download and upload), or indirectly through use of previously generated mini disks (for stand-alone PCs). Diagnosis of behavior in the PMDSS runs will be done through use of the Projection Analyzer.

All user access to the PMDSS will be controlled by the User Interface. Specific selections will generate commands to the other modules for execution of specific analytical tools. The User Interface is the interactive window which enables the user to provide inputs to the projection model and to view the results of that model. It is similar to functions provided by the Management Information Systems (MIS) that are tied to production systems within Alcott..

Most production projection models run in the MVS environment. Interface modules must be added at MVS environment to provide the baseline data needed by the PMDSS. This data will be stored on the VM mainframe for downloading to each PC that incorporates the PMDSS. Using the VM mainframe as a data repository will help insure that all user PCs have the latest version of data.

The PMDSS will generate a set of output data for each alternative scenario projected. The option will exist to upload this data to the VM mainframe so that it may be accessed by the other PMDSS users.

Each user's PMDSS will also have the built-in capability to display and compare baseline alternatives and PMDSS alternatives, either graphically or in tabular reports. But the number of alternatives stored in the PMDSS will be limited by the disk storage on the user's PC.

The user will have the option of developing his own tailored applications, based on the baseline data stored in the PMDSS data base. These applications include:

a. spreadsheets

b. statistical analysis procedures

c. tabular reports/ graphics

d. optimization models

The PMDSS will be designed for the complete PC environment. The objective is not just to build specialized tools for the analysis of data, but also to provide an interface to the best off-the-shelf tools for use by the individual user in building his own applications[30] .

3.2.1 PC Environment

The major components of the PMDSS on the user's PC are:

- User Interface
- PM Modules
- Model Manager
- Data Base Manager
- PMDSS/VM Interface
- Projection Analyzer

The Projection Analyzer is not discussed further in this study, as its exact form has not yet been determined. It will be researched further as development proceeds, and as we obtain a clearer understanding of the "why" questions that are important to the user. The "why" questions will surface as the prototype and follow-on versions of the PMDSS are used by the Program Managers.

---

[30] Ralph H. Sprague, Jr. And Hugh J. Watson, Decision Support Systems: Putting Theory Into Practice (Prentice Hall, Englewood Cliffs, New Jersey, 1986), p. 98.

User Interface

The user interface controls all aspects of the PMDSS. The flow of the PMDSS interface is depicted in Figure 3.2.1-1. This figure shows the steps the user would take to:

a. View, input or modify projection input data. These data are modified by the user to reflect a particular scenario.

b. View or output projection results.

The following paragraphs describe how the user could navigate through the user interface. This is a description for the purposes of the concept study; the design of the actual PC menus and displays used in the PMDSS will be reviewed by users during the early stages of development of the initial prototype. The description user interface will be supportable by the minimum PC configuration.

Select Module

The user will "boot-up" his PC, and then enter "PMDSS" at the DOS prompt line. The first menu is the **Select Module** menu. This allows the user to enter the functional module he wishes to work at:

- Critical Factors
- R & D Programs
- BP Programs
- MP Programs
- Executive Support

The user will work with only one module at a time. However, he will be able to move quickly from one module to another without leaving the PMDSS. The PM Modules and how they interact are discussed further in "The PMDSS Modules" section.

Select Scenario

Once a module is selected, the user may pick previously entered projection scenarios to work from in the **Select Scenario** menu.

An scenario is defined by its "building blocks":

- Input scenarios
- Production baselines
- Interfacing scenarios

A scenario run creates a unique set of projection results which may be viewed by the user.

The scenario definition may be viewed, modified, copied or deleted from the PMDSS DB.

Define Scenario

If a user wishes to change the building blocks of a scenario, he enters the **Define Scenario** menu. In this menu he can:

- Copy a particular class of inputs from another scenario. This data can then be modified under the current scenario. The scenario copied from may be a production run (i.e., baseline) or another PMDSS run.

Import/Export Data

The **Import/Export Data** menu will be used to bring in PMDSS alternatives from DOS files outside the PMDSS DB. These files will have been transferred from the VM mainframe PMDSS DB or from floppy disk. This menu will also allow the PMDSS alternative to be written to external DOS files.

Initially the transfer of files to the PMDSS will be accomplished in two operations:

- For importing alternatives: first the user will transfer the files from the floppy disks or from the VM maintenance to DOS, and then enter the PMDSS to import the files.
- For exporting alternatives: first the user will export the files from the PMDSS to DOS, and then exit the PMDSS to transfer them to floppy disks or VM mainframe.

Later the process to transfer data to/from floppies can be upgraded to be done within the PMDSS. The process to transfer data to or from the mainframe from within the PMDSS may also be added later. However, this enhancement may not be supportable under the 640 Kb memory in the minimum PC platform.

Scenarios may be passed from user to user, either through floppy disks or through the PMDSS <->VM interface. This will allow one user to set up part of a projection and then pass it to a co-worker so that he can add his input and execute the projection. In addition, the ability to export an entire alternative to floppy disk or the VM maintenance will provide a back-up to the PC data. The data could then be easily restored if the data were destroyed on the PC.

Select Display Template

Once the user has selected and/or defined the alternatives he wishes to work with, he enters the **Select Display Template** menu. A display template defines a set of input data and/or output results that the user wishes to select. Conceptually, a display template is a window on the input and output that the user wishes to see. The template will make the PMDSS much more user-friendly than the production systems. If the user typically wants to look at aggregate net profit, he can preselect that template which he has created earlier. He won't have to navigate through a number of menus each time he enters the PMDSS to display that information.

In the **Select Display Template** menu he can select one or more display templates to build his "window" on the data for that session. A template is not tied to a particular scenario; it is only tied to a specific functional module. If desired, he may copy, delete, add to or create a template. In addition, the user may view the template, and if desired, he can change the order of data in the template to allow easier comparisons of similar data. Templates have user-assigned names and text definitions, as do alternatives. Each template will have an upper limit on the number of time-series it specifies (at least 100).

If the user wishes to create a new template or add to an existing one he enters a series of menus to select the specific data that he wishes to include in the template. These template selection menus will also lead the new or infrequent user to his desired data directly, without using the template-driven "fast path".

A user may go down the three template selection menus several times, to build a set of data to be displayed in a step by step fashion. The active selections may be

displayed at any time during this selection process. This is how a user would build a template with multiple data types, dimension combinations, etc.

Select Dimensions

The first of these template selection menus is **Select Dimensions**. Here the user will choose the specific combination of dimensions he wishes to access, such as:

- Product
- Total Business
- Customer
- Product Category
- Geographical Region
- Business Segments

Select Dimension Value

The user will then enter the **Select Dimension Values** menu. In this menu the user will select the specific values for the dimension he has chosen. For example, the user may select several or all business segments (hospitals, government, retail) in the **Select Dimensions Values** menu.

Select Variables

The user will then enter the **Select Data Variable** menu. Here the user will choose the specific data types or variables he wishes to work with under the chosen dimensions. All data types available for a dimension will be selectable in the menu.

Data Interface

After making selections on these three menus, the user has defined a template or has selected data for the current session. He then enters the **Data Interface** display. This display is where all scenario data is displayed and worked with, whether for input to the projection or for projection output.

Both input and output data can be displayed simultaneously. Input data is not segregated from output data because some data may be both an input to the projection or an output of the projection. The user may scroll left or right in the **Data Interface** display to view different time-series, or scroll up or down to see different time ranges. Data may be displayed either by month, quarter or by year for projection scenarios, and by month to date (MTD), quarter to date (QTD) and year to date (YTD) for curent status scenarios.

The **Data Interface** display is the most important part of the user interface. This display will allow the user to view the critical inputs he wishes to see, the effects of changing, and the important outputs. The user will be able to save the alternative with all of his updates through this panel as well. This display will also allow the user to display the input and/or output and/or output data from several scenarios simultaneously.

Two other menus will be used to provide the user with additional options. These options are set by the user and saved from PMDSS sessions to session. These menus are:

a. Display options

b. Report options

These menus are discussed below.

Display Options

The **Display Options** menu will let the user modify parameters such as:

- The time periods to be displayed: quarter or years.
- The time interval to be displayed, in years. Up to 20 years of data may be displayed, if the data exists.
- Whether "deltas" are to be displayed or not.

Report Options

The **Report Options** menu will let the user choose to report:

- Graphical or tabular data.
- To the screen, a file or a printer.
- If to a file, as a formatted ASCII report, in unformatted ASCII, or as a LOTUS formatted file.
- With specific report formats chosen by the user.

The report generator will initially be a way to dump the input and/or output on the **Data Interface** display to a file or printer. Formatting options will be added later.

General User Interface Features

The user Interface will have features for the novice user and for the expert who can exploit "fast path" options such as the use of display templates. Other general features will include:

- A context-specific HELP facility which can be reached from each menu/display.
- A "quick-exit" facility for exiting the PMDSS from any point, while also allowing the user to save his changes, if desired.
- Standard PF keys
- Standard Menu Colors

Model Manager

This module will interpret data and commands from the User Interface and execute the appropriate mathematical models or procedures. The model manager will automate the model execution process, which will be entirely on the user's PC. Models may be executed immediately after all input modifications are made by the user.

A facility to run a number of projections "in batch" may also be desired at a later date. The user could set up a number of alternatives and then run these during a time when he does not need to use his PC for several hours. A batch facility will also become more useful if the PC Operating System is upgraded to allow multiple sessions (i.e., from DOS to Microsoft Windows/DOS or OS/2). In this case the user could work on other tasks in the "foreground" session s while the projections are running in the "background" session.

PMDSS <-> VM Interface

The PMDSS will export scenarios stored on its PC DBMS to DOS Files. These scenarios will then be transferred to CMS "Flat" Files, where a "Flat" File is defined to be a sequential file not part of a file or data base management system (DBMS), through

the PC mainframe DBMS. The reverse process will be from the VM DB to the PC DB. Each step will be performed through use of simple menu selections or commands.

Interfaces with Other PC Applications

The PMDSS will produce formatted ASCII or LOTUS Files which may be used by external PC applications. These applications include statistical analysis software, spreadsheets and even word processors. Some work by the analyst will be needed to import the PMDSS data into the application but it will be relatively minor.

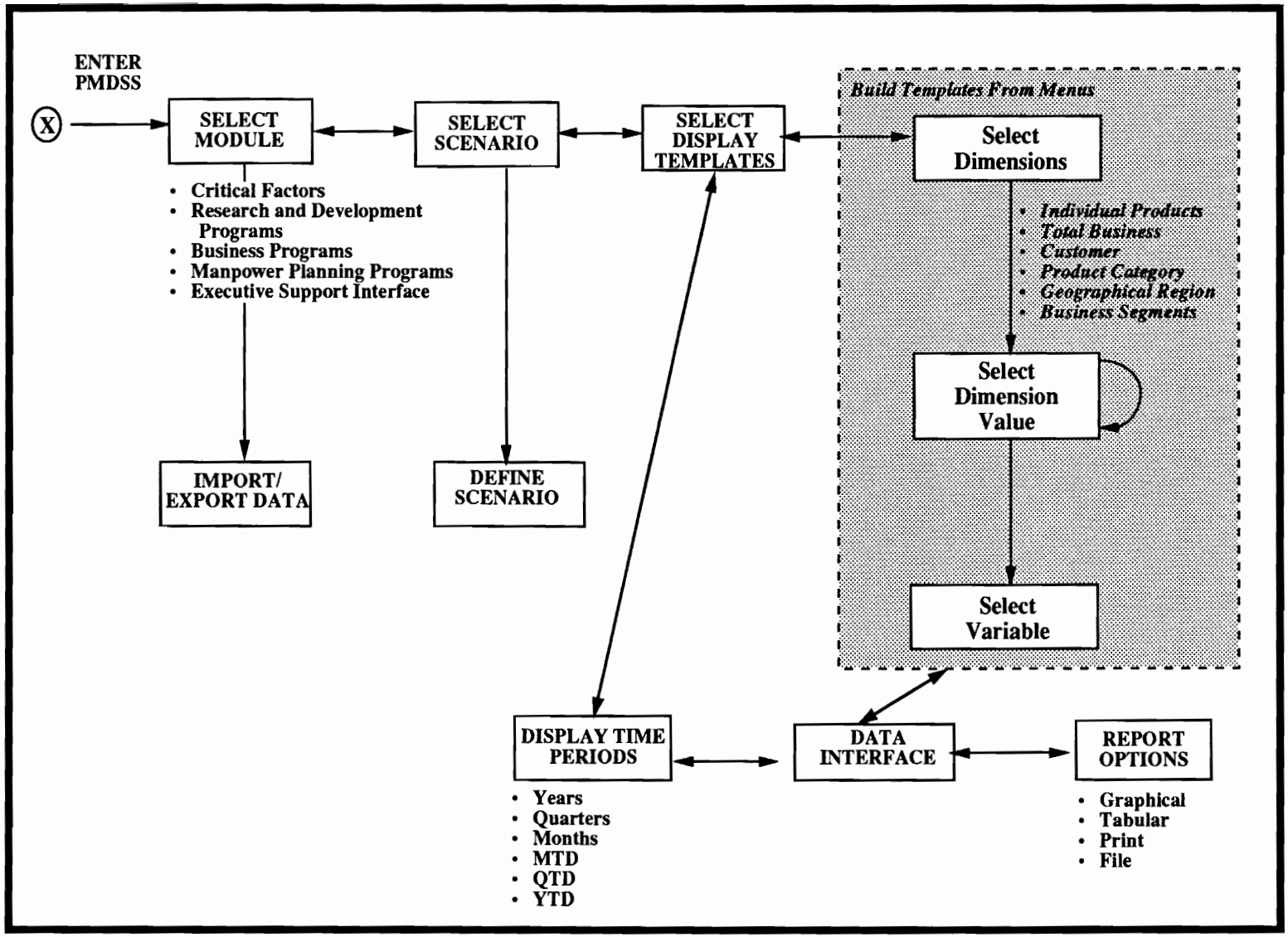


Figure 3.2.1-1. PMDSS User Interface Functions

3.2.2 VM Environment

The VM environment will contain:

- a shared PMDSS DB
- an interface with other VM Applications
- a VM <->MVS interface

Shared PMDSS Databases

Each user will operate the PMDSS on his own PC. The PMDSS will be designed so that a single user can manage the task of putting together and executing an alternative scenario without extensive computer expertise. However, there will be instances where several users would want to pool their efforts in building a projection. Perhaps one user changes budget data while another manages cost data. Their inputs would have to be combined to make up a complete projection scenario.

Sharing or archival of all PMDSS alternatives will be done through use of a single DB on the VM environment . This data base will be under the Structured Query Language/Data Base (SQL/DB), the primary relational DBMS used on the VM. A DB design similar to that used in the PMDSS PC DB will be used.

The user will be able to make SQL queries on the PMDSS data through the VM. In addition, utilities will allow the data to be read from or dumped to CMS flat files for use in other applications.

Interface with Other VM Applications

The user will be able to extract desired data from the PMDSS SQL DB for use in external programs such as Easytrieve, a report generator.

VM <-> MVS Interface

The MVS interface files will be created as a sequential (i.e., Flat) files on the MVS computer, and then copied to the VM through a batch job. Finally, the VM-Resident Flat Files will be reformatted and loaded into the SQL DS.

3.2.3 MVS Environment

The transactional systems and production projection models reside on the MVS computer. This data will provide some of the baseline information needs. Separate job steps will be added to each of these systems to produce the outputs required by PMDSS.

The exact design and development of the interfaces required from the production systems will be significant part of the overall PMDSS effort. Some of the data required for projection is internal to the modeling process of some of the projection models, and cannot just be pulled from a convenient source like the VM MIS.

3.3 PMDSS Modules

The PMDSS Modules are a network of interconnected models and procedures that incorporate the functional needs of the executive team in 3 major areas (Business Planning, R&D and Manpower Planning). Each module is oriented to performing particular functions for that area. Data may be passed from one level to another. Each module will be discussed in more detail in the next section.

3.3.1 Critical Factors Module (CFM)

No one, including the executive, knows all the critical factors within his organization. The CFM within PMDSS attempts to identify a few for now, leaving the remainder to evolve as the users use PMDSS

- operating results (cost vs revenue)
- program status (budget vs actual)
- profitability execptions (which product is below "par", what is "par")

at a particular point in time (MTD, QTD, YTD) for any dimension (product, region) available to select from[31] .

The PMDSS CFM will selectively extract data from real-time operational and transactional systems on a daily basis, as well as extract data from existing strategical, tactical and operational models when they are updated. One major objective of the CFM

---

31 Efraim Turban, Decision Support and Expert Systems: Management Support Systems (Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990), p. 391.

is to serve as an initial point of analysis for the Operations Research (OR) analyst and as a data driver for the data base (DB) analyst. The OR analyst must, for example ensure that the data obtained in a 5-10 year model of projected sales is used as input to a model conveying projections by month, or data obtained in a 5-10 year model of projected funds to be budgeted are used as constraints to a model allocating available funds. In otherwords, the same data must flow from one model to another, with consistency through out all the dimensions identified. To achieve this, the OR analyst may have to adjust existing models and the DB analyst may have to revise existing data extraction routines or even derive an integrated data base. Existing inputs and outputs of models and reports will be analyzed by the OR and DB analyst to determine other variables which influence Alcott's critical success factors.

Another CFM objective is to provide consistency in measuring performance. The PM no longer has to wait for his annual review to determine how well he is performing; PMDSS tracks his performance on a daily basis and gives him the opportunity to investigate and analyze his results.

3.3.2 Research and Development Programs Modules

This section describes 5 cases (scenarios) using 2 mathematical techniques: Dynamic Programming and Risk Analysis Simulation.

Dynamic Programming Modules

Dynamic programming models will be a subset for the R&D modules. Dynamic programming is a general technique for solving problems involving a set of interrelated decisions in which the goal is to optimize overall effectiveness[32] . This method is used to break larger problems down into smaller subproblems, where the smaller problems are solved sequentially until the solution of the original problem is reached. In the process of solving the smaller sub-problems, the decision maker uses the solutions to other subproblems already obtained at the previous stage. Thus the subproblems do not have to be solved "From Scratch" but are solved by using previously solved sub-problems. This will be illustrated in the examples below[33] .

R&D Model Base Case 1

To illustrate the dynamic programming formulation, consider the following R&D Model Case 1: A fixed amount of money is available to allocate in integral units to various R & D programs (projects).

---

[32] Larry M. Austin And James R. Burns, Management Science: An Aide for Managerial Decision Making (Macmillan Publishing Company, New York, Collier Macmillan Publishers, London, 1985), p. 412.

[33] Michael Q. Anderson And R. J. Lievano, Quantitative Management: An Introduction (Second edition. Kent Publishing Co., Boston, Massachusetts, 1986), pp. 567-570.

Given the following variable definitions:

R_i ~ return from program i

A_i ~ resource allocated to program i

T ~ total amount of resources available

N ~ number of programs

and given the following table of input data.

Input Data

Program	A_i	R_i	i
P1	200	20	1
P2	250	35	2
P3	320	56	3

where T = 1000, Goal: Compute $F_1^*(1000)$

Determine the optimal allocation in each program to maximize total return.

The following definitions apply:

- i ~ stage (number of programs selected)
- s_i ~ state (status of the system)
- d_i ~ decision at stage i
- f ~ return function (measure of effectiveness like cost, profit, etc.)
- $f_i(s_i, d_i)$ ~ processing optimally at stage (i+1)
- $f_i^*(s_i)$ ~ optimal value of f_i (optimal return function)
- d_i^* ~ optimal decision at stage i in state s_i.

The formulation steps are as follows:

1. Formulate as a Dynamic Program
2. Max $Z = R_1(A_1) + R_2(A_2) + \ldots + R_N(A_N)$
   S.T. $A_1 + A_2 + \ldots + A_N \leq T$

Let i ~ Stage

s_i ~ state ($)

amount of money left for allocating at the beginning of stage i

Figure 3.3.2-1 illustrates the User Interface for Case 1. We formulate the problem as a three stage program because the user selects 3 programs. At Stage 3 we decide how many $320 P3 units to invest in; at Stage 2, we decide how many $250 P2 units to invest in; at State 3, we decide how many $200 P1 units to invest in. The stage s_i is defined as the amount of money left for investment at the beginning of state i.

Stage 3 R&D Model Case 1

Table 3.3.2-1 represents the results of computing f_3 and f_3^*, where s_3 indicates the ranges of the stage variable corresponding to these 4 ranges, the feasible values for decision d_3 are:

d_3	RANGE
0	1
0,1	2
0,1,2	3
0,1,2,3	4

$f_3(s_3, d_3) = R_3d_3 + f4 * (s_3 - a_3d_3)$

$= 56d_3 + 0$

$= 56d_3$

$f_4^* = 0$ because Stage 3 is the last stage and there is nothing left.

$f_3^*(s3) = \max \{f_3(s_3, d_3)$: all feasible $d_3\}$
$= \max \{56\ d_3$: all feasible $d_3\}$

Stage 2 R&D Model Case 1

Table 3.3.2-1 answers the following questions. How much should be invested in P2 to maximize our yearly return over stages 2 and 3. To better understand the entries in the table, let's consider the following: Suppose we have \$800 to invest in P2, and each unit investment is \$250, then the feasible value of the decision variable d_2 (number of investments) are 0, 1, 2, 3 thus

$f_2(800, d_2) = R_2\ d2 + f_3^*(800 - 250\ d_2)$
If $d_2 = 0$, $f_2(800,0) = 35(0) + f_3^*(800) = 0 + 112 = 112$
If $d_2 = 1$, $f_2(800,1) = 35(1) + f_3^*(550) = 35 + 56 = 91$
If $d_2 = 2$, $f_2(800,2) = 35(2) + f_3^*(300) = 70 + 0 = 70$
If $d_3 = 3$, $f_3(800,3) = 35(3) + f_3^*(50) = 105 + 0 = 105$

Thus
$$f_2^*(800) = \max \begin{bmatrix} 112 & , d_2 = 0 \\ 91 & , d_2 = 1 \\ 70 & , d_2 = 2 \\ 105 & , d_2 = 3 \end{bmatrix}$$

$= 112$ with $d_2 = 0$

$f_2^*(800) = 112$ with $d_2^* = 0$ can be interpreted as follows:

If we end up at the start of Stage 2 with \$800, we should invest \$0 in P2 and carry the \$800 to Stage 3 (P3).

Stage 1 R&D Model Case 1

The goal of the problem is to compute f_1^* (1,000), that is the maximum yearly return over stages 1, 2 and 3 (the 3 program alternatives) if we have 1,000 to invest. Thus s_i takes on the single value 1,000. Since stage 1 consists of unit investments of \$200 each, d_1 may equal 0, 1, 2, 3, 4 or 5. $[0 \leq d_1, \leq s_1, / 200]$.

$f_1^*(1{,}000) = 168$ with $d_1^* = 0$.

Optimal Policy

$s_1 = 1{,}000 \quad d_1^* = 0 \quad f_1(1{,}000) = \$168$
$s_2 = 1{,}000 \quad d_2^* = 0$
$s_3 = 1{,}000 \quad d_3^* = 0$

Allocate 0 units to P1
0 P2
0 P3
Retain \$40 cash

Annual return = \$168.

R&D Model Case 2

Suppose the user wishes to modify the base case to include a 4th investment alternative allocated at \$100 and yielding an annual rate of \$14. The following table represents the modified input:

PROGRAM	A_i	R_i	i
P1	100	14	1
P2	200	20	2
P3	250	35	3
P4	320	56	4

* T = 1000

Goal: Compute $f_1^*(1{,}000)$

Figure 3.3.2-2 illustrates the User Interface for this example. The 4th program investment has been placed ahead of the existing 3 so that the table corresponding to P3 and P2 of the base case 1 need not be recalculated for this case. That is Stage 3 of the base case is equal to Stage 4 of this example and Stage 2 of the base case is equal to Stage 3 of this example. This demonstrates 1 advantage in dynamic programming.

Tables for Stage 4 - Stage 1 are depicted in Table 3.3.2-2.

$s_1 = 1{,}000 \quad d_1^* = 0$

$s_2 = 1{,}000 \quad d_2^* = 0 \qquad f_1^*(1000) = \$168$

$s_3 = 1{,}000 \quad d_3^* = 0$

$s_4 = 1{,}000 \quad d_4^* = 3$

Optimal Policy

Allocate 0 units to P1
Allocate 0 units to P2
Allocate 0 units to P3
Allocate 3 units to P4
Retain $10 Cash

R&D Model Case 3

Suppose, the investment strategy has changed from the base case because the user only has $900 to invest instead of $1000. The solution can be found in the Stage1 table for the base case, while the other tables do not need to be changed. This demonstrated another general property of dynamic programming; it is often very easy to perform sensitivity analysis.

The User Interface is illustrated in Figure 3.3.2-3 and the Table for computing the New Stage 1 is in Table 3.3.2-3 and was derived as follows:

$$f_1(s_1,d_1) = 20\,d_1 + f_2^*\,(s_1 - 200\,d_1)$$

$$f_1^*\,(900)\ \max\ \{20d_1 + f_2^*(900 - 200d_1):\ \text{all feasible } d_1\}$$

Optimal Policy

$s_1 = 900 \qquad d_1^* = 0$
$s_2 = 900 \qquad d_2^* = 1 \qquad f_1^*(900) = \$147$
$s_3 = 650 \qquad d_3^* = 2$
Allocate 0 units to P1
1 units to P2
2 units to P3
Retain $10
Annual Return = $147

R&D Model Case 4

Suppose the user would like to know if his/her investment strategy would change if the return of P1 was $36 instead of $20. The user interface is illustrated in Figure 3.3.2-4, and the New Stage 1 Table is depicted in Table 3.3.2-4 and was derived as follows:

$$f_1(s_1, d_1) = 36d_1 + f_2^*(s_1 - 200d)$$

$$f_1(1{,}000) = \max \{ 36\, d_1 + f_2^*(1{,}000 - 200 d_1) : \text{all feasible } d_1 \}$$

Optimal Policy

$s_1 = 1000 \quad d_1^* = 5 \quad f_1^*(1{,}000) = 180$

$s_2 = 0 \quad d_2^* = 0$

$s_3 = 0 \quad d_3^* = 0$

Allocate 5 units to P1
0 units to P2
0 units to P3

Retain $0 cash

Annual return = $180

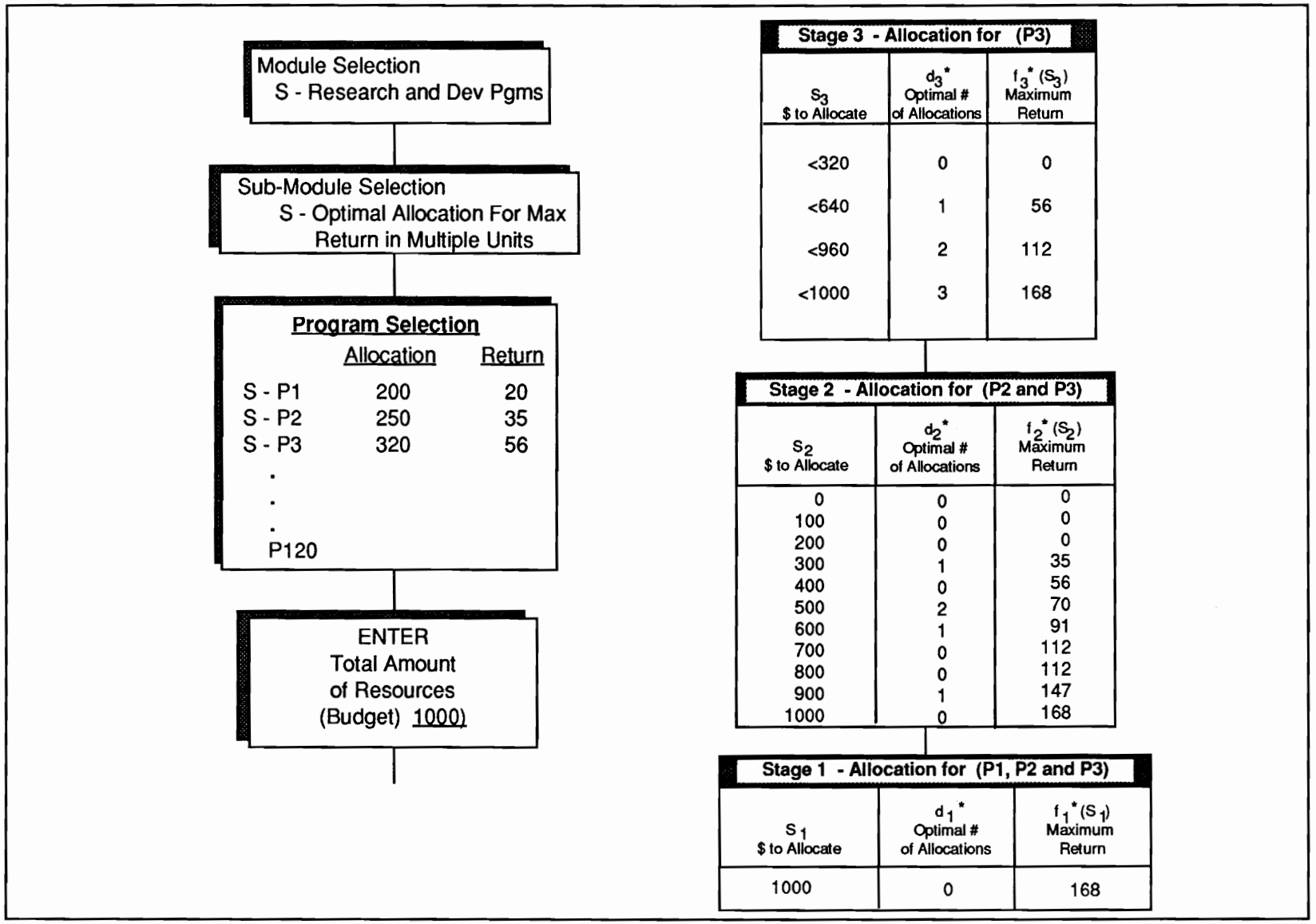
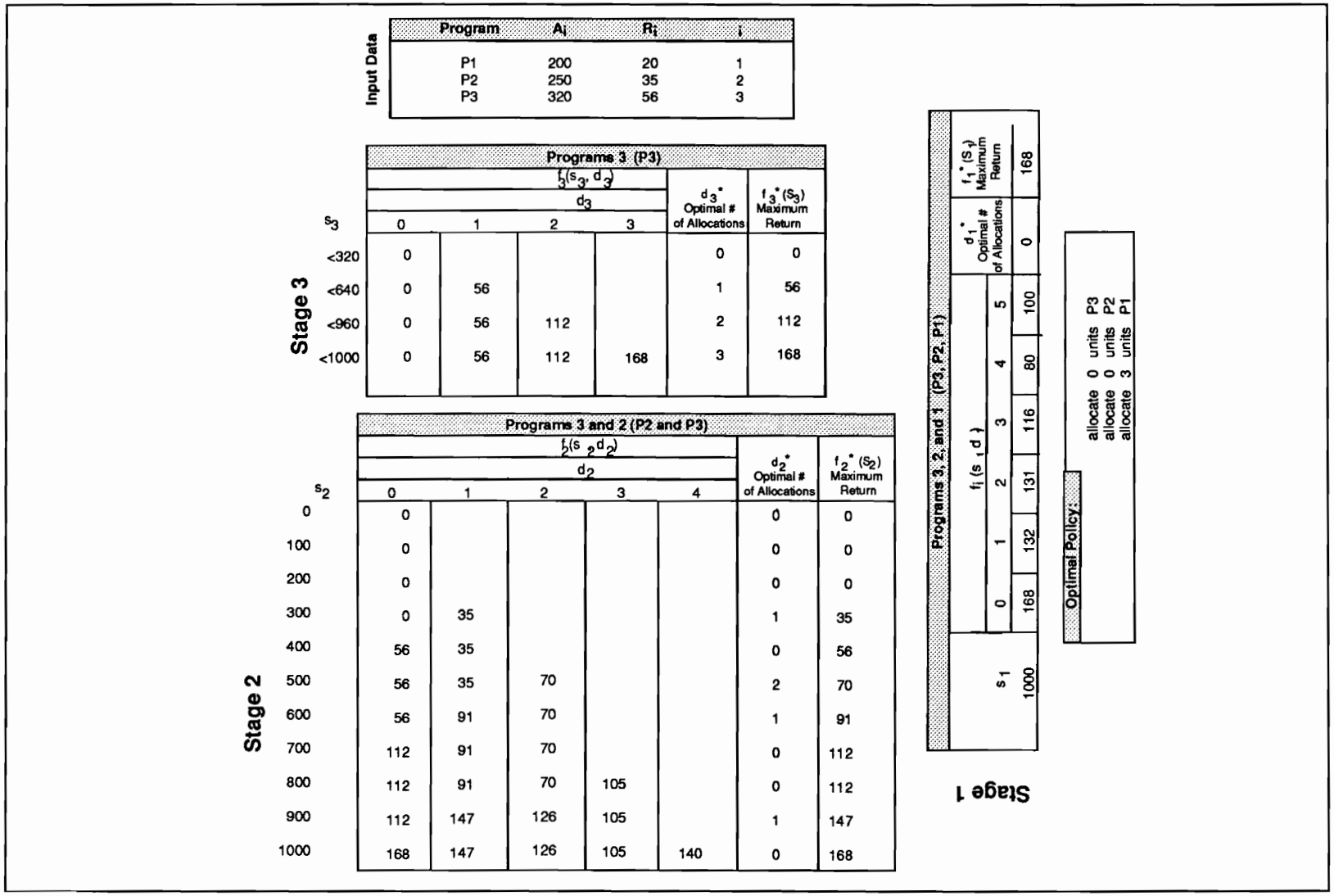


Figure 3.3.2-1. User Interface For R&D Dynamic Programming Model Case 1

Table 3.3.2-1. Table For Computing R&D Dynamic Programming Model Case 1



Input Data

Program	A_i	R_i	i
P1	200	20	1
P2	250	35	2
P3	320	56	3

Stage 3

Programs 3 (P3)

s_3	$f_3(s_3, d_3)$ d_3 = 0	1	2	3	d_3^* Optimal # of Allocations	$f_3^*(S_3)$ Maximum Return
<320	0				0	0
<640	0	56			1	56
<960	0	56	112		2	112
<1000	0	56	112	168	3	168

Stage 2

Programs 3 and 2 (P2 and P3)

s_2	$f_2(s_2 d_2)$ d_2 = 0	1	2	3	4	d_2^* Optimal # of Allocations	$f_2^*(S_2)$ Maximum Return
0	0					0	0
100	0					0	0
200	0					0	0
300	0	35				1	35
400	56	35				0	56
500	56	35	70			2	70
600	56	91	70			1	91
700	112	91	70			0	112
800	112	91	70	105		0	112
900	112	147	126	105		1	147
1000	168	147	126	105	140	0	168

Stage 1

Programs 3, 2, and 1 (P3, P2, P1)

s_1	$f_i(s_1 d_i)$ d_i = 0	1	2	3	4	5	d_1^* Optimal # of Allocations	$f_1^*(S_1)$ Maximum Return
1000	168	132	131	116	80	100	0	168

Optimal Policy:

allocate 0 units P3
allocate 0 units P2
allocate 3 units P1

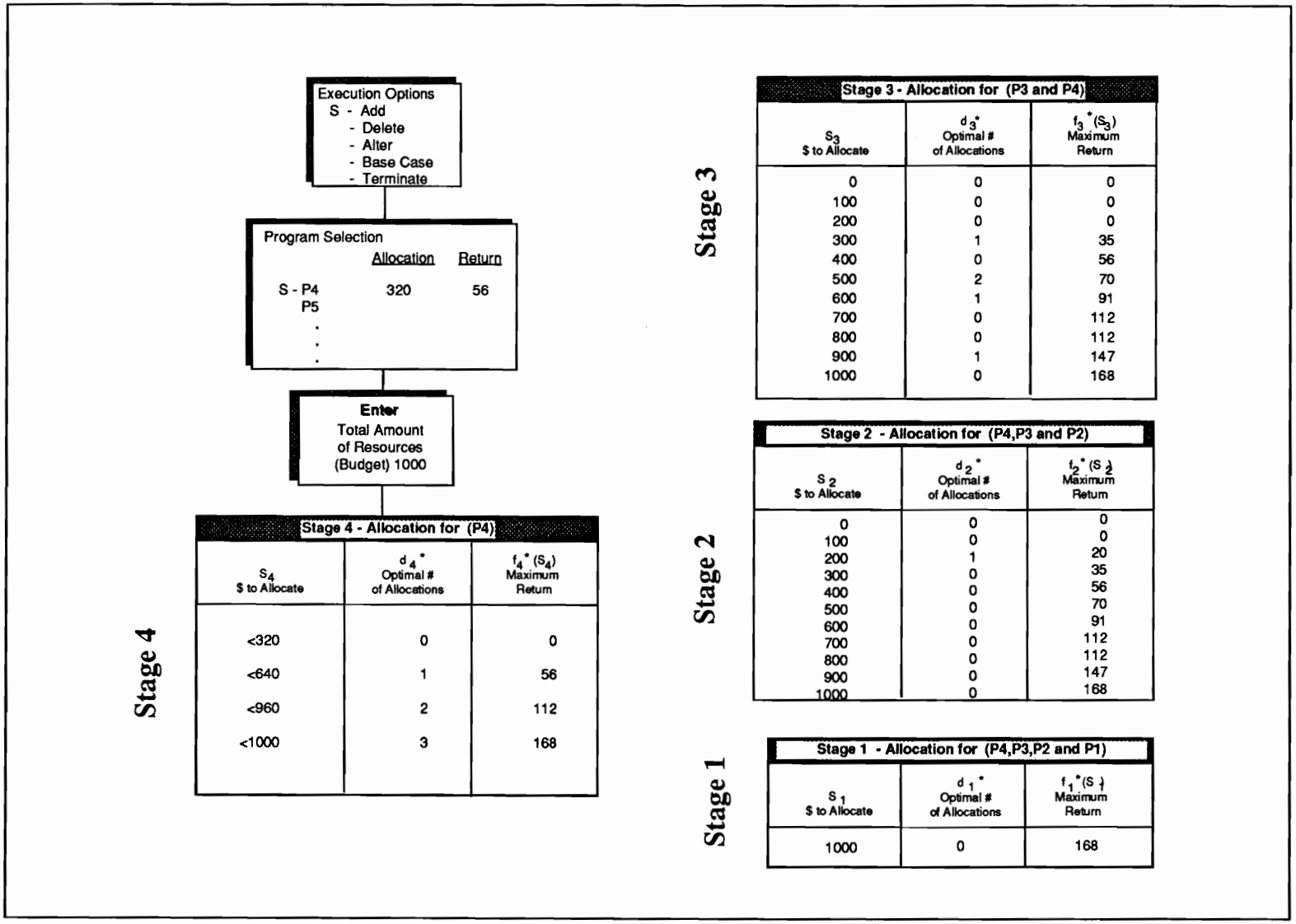


Figure 3.3.2-2. User Interface for R&D Dynamic Programming Model Case 2

Table 3.3.2-2. Table for Computing R&D Dynamic Programming Model Case 2

Stage 4 - Program 4 (P4)

Stage 4	s_4	$f_4(s_4, d_4)$ d_4 = 0	1	2	3	d_4^* Optimal # of Allocations	$f_4^*(S_4)$ Maximum Return
	<320	0				0	0
	<640	0	56			1	56
	<960	0	56	112		2	112
	<100	0	56	112	168	3	168

Stage 3 - Program 4 and 3 (P4,P3)

Stage 3	s_3	$f_3(s_3, d_3)$ d_3 = 0	1	2	3	4	d_3^* Optimal # of Allocations	$f_3^*(S_3)$ Maximum Return
	0	0					0	0
	100	0					0	0
	200	0					0	0
	300	0	35				1	35
	400	56	35				0	56
	500	56	35	70			2	70
	600	56	91	70			1	91
	700	112	91	70			0	112
	800	112	91	70	105		0	112
	900	112	147	126	105		1	147
	1000	168	147	126	105	140	0	168

Stage 2 - Program 4,3,2 (P4,P3 and P2)

Stage 2	s_2	$f_2(s_2, d_2)$ d_2 = 0	1	2	3	4	5	d_2^* Optimal # of Allocations	$f_2^*(S_2)$ Maximum Return
	0	0						0	0
	100	0						0	0
	200	0	20					1	20
	300	35	20					0	35
	400	56	20	40				0	56
	500	70	55	40				0	70
	600	91	76	40	60			0	91
	700	112	90	75	60			0	112
	800	112	111	96	60	80		0	112
	900	147	132	110	95	80		0	147
	1000	168	132	131	116	80	100	0	168

Stage 1 - P4, P3, P2, P1

Stage 1	s_1	$f_1(s_1, d_1)$ d_1 = 0	1	2	3	4	5	d_1^* Optimal # of Allocations	$f_1^*(S_1)$ Maximum Return
	1000	165	161	140	154	147	140	0	168

Stage 1 - P4, P3, P2, P1

Stage 1	s_1	$f_1(s_1, d_1)$ d_1 = 6	7	8	9	10	d_1^* Optimal # of Allocations	$f_1^*(S_1)$ Maximum Return
	1000	140	133	132	126	140	0	168

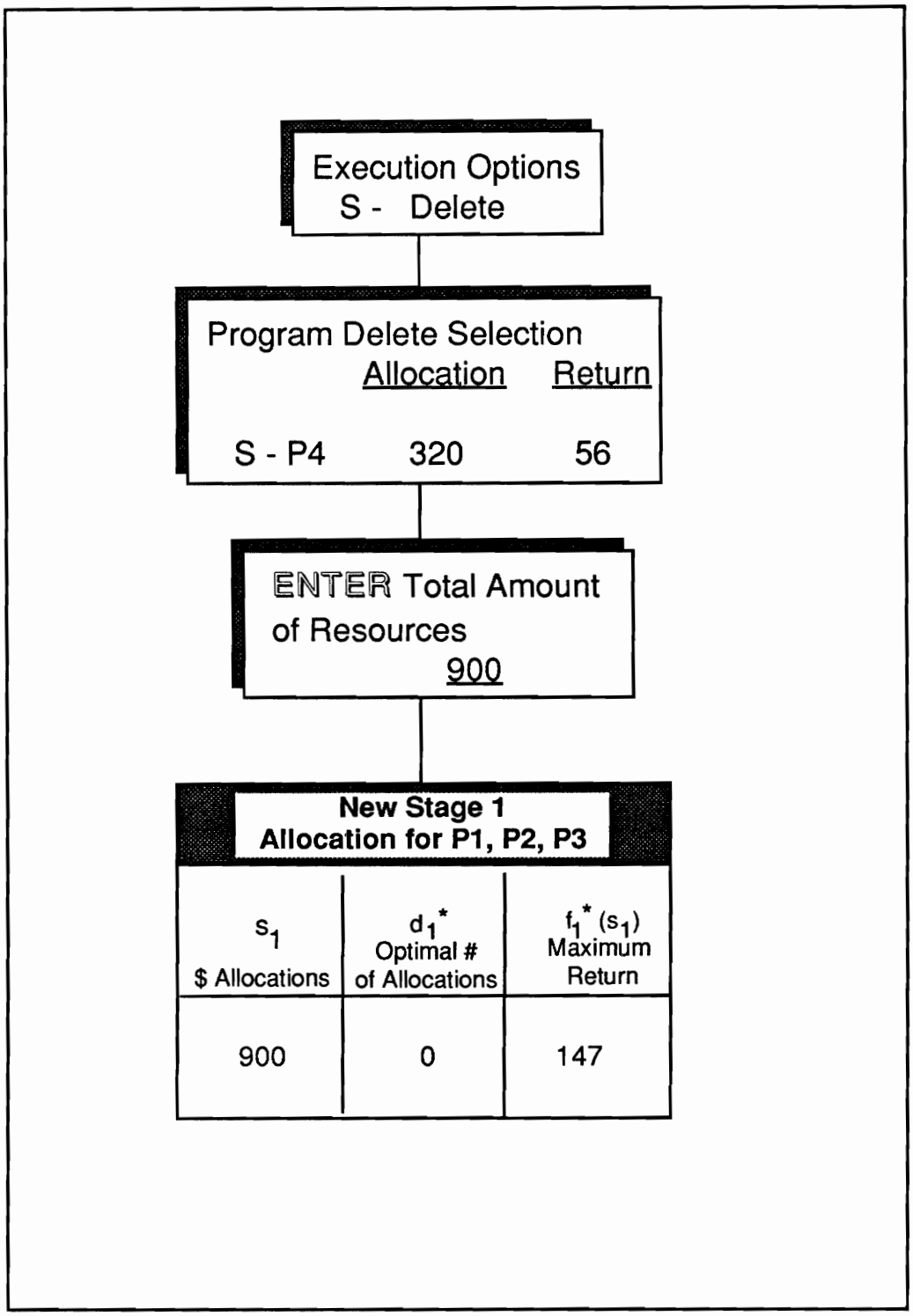


Figure 3.3.2-3. User Interface For R&D Dynamic Programming Model Case 3

Table 3.3.2-3. Table For Computing R & D Dynamic Programming Model Case3

	New Stage 1 - P3, P2, P1						
	$f_1(s_1, d_1)$						
	d_1					d_1^* Optimal # of Allocations	$f_1^*(s_1)$ Maximum Return
s_1	0	1	2	3	4		
900	147	132	110	95	80	0	147

Optimal Policy

Allocate 0 units to P1

Allocate 1 units to P2

Allocate 2 units to P3

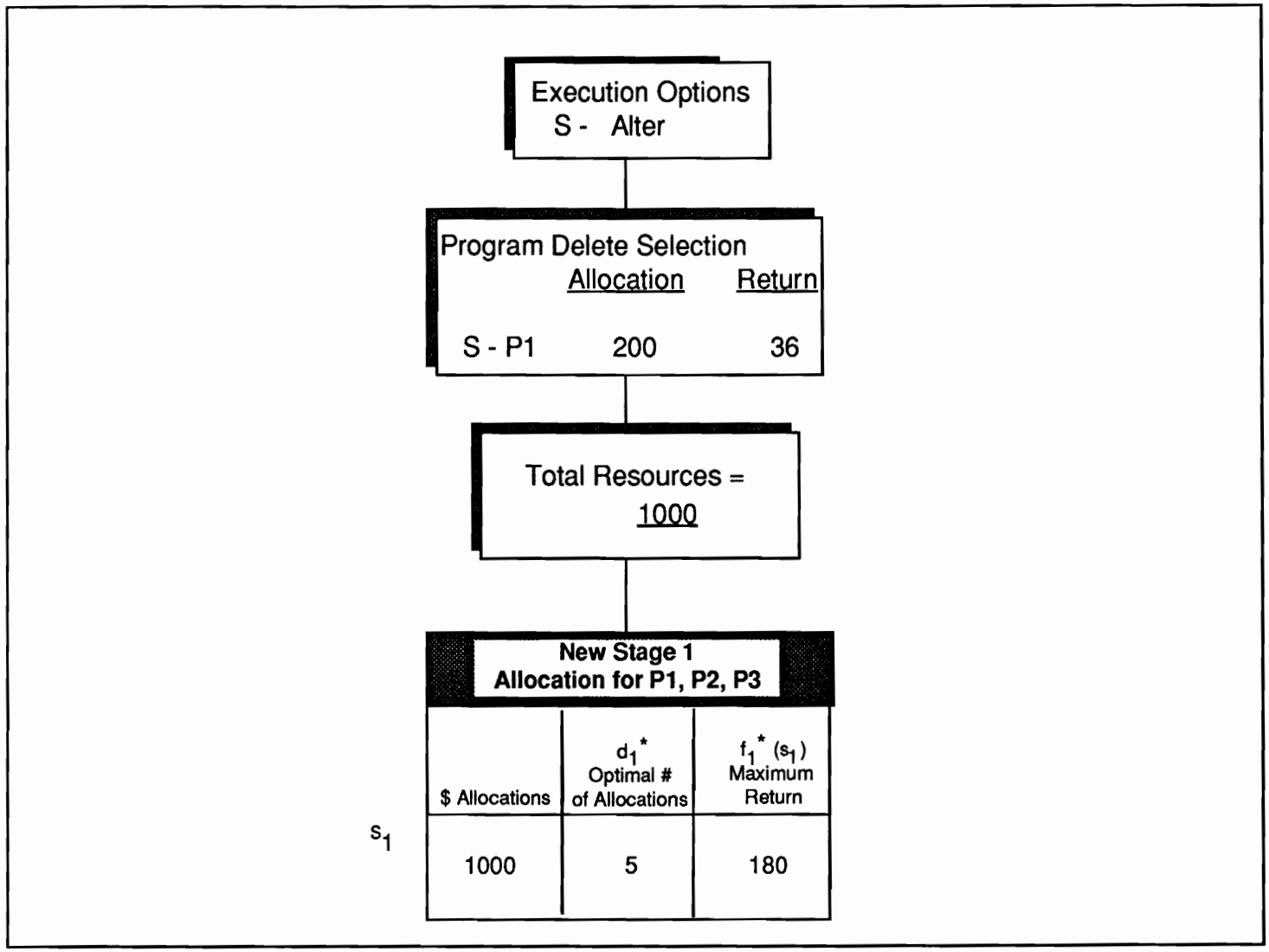


Figure 3.3.2-4. User Interface For R & D Dynamic Programming Model Case 4

Table 3.3.2-4. Table For Computing R & D Dynamic Programming Model Case 4

New Stage 1

	New Stage 1 - P3, P2, P1							
	$f_1(s_1, d_1)$						d_1^* Optimal # of Allocations	$f_1^*(s_1)$ Maximum Return
	d_1							
s_1	0	1	2	3	4	5		
1000	168	148	163	164	144	180	5	180

Optimal Policy
Allocate 5 units to P1
Allocate 0 units to P2
Allocate 0 units to P3
Retain $0 Cash

R&D Programs Using Risk Analysis, Case 5[34]

Program managers must often decide whether to make capital investments. For example, a new computer may allow Alcott to introduce a new product category. The additional profit must be compared with the equipment cost to decide whether this investment is worthwhile to Alcott. One way to analyze this situation is to consider the time value of money (net present value (NPV) Analysis) where the capital investment is paid at the beginning, but the profit is realized over its useful life. The cash flow during the life of the new computer must be estimated. Outflows of cash are negative, and inflows of cash are positive. In NPV analysis, all cash flows are discounted back to the beginning at some specified discount rate to calculate the NPV. The decision here is whether or not to make the capital investment. If the NPV is positive, the capital investment is attractive; if the NPV is negative, the capital investment is unattractive[35] .

One approach for considering the uncertain factors (annual sales, useful life), is to explicitly incorporate their probability distribution into the analysis. The risk analysis technique uses probabilistic simulation to estimate the NPV. You determine the probability distribution for each uncertain factor, generate random variables from those distributions, and calculate the NPV for repeated experiments. You use the simulation results to estimate the NPV and its probability distribution.

Let's consider an example:

Assumptions:

---

[34] Thomas W. Knowles, Management Science, Building And Using Models (Irwin, Homewood, Illinois, 1989), pp. 873-878.

[35] David Whitaker, OR on the Micro (John Wiley and Sons, Whichester, 1984), p. 129.

- Selling Price (SP) and Operating Costs (OC) are good, so these variables can be treated as deterministic
- Annual Sales (AS), SP, and OC stay the same from year to year
- AS and Useful Life (N) are independent random variables.

Let

SP = \$500 / unit
OC = 300 / unit
SV = 0
CC = \$1 million

Where SV = Salvage Value of the equipment after the end of its useful life (N)
CC = Capital Cost

From the historical database, the probability distributions can be estimated for the sale ranges and useful life categories indicated below as follows:

(AS) Sales	Probability	Cumulative Probability
1,000	0.2	0.2
1,200	0.5	0.7
1,400	0.3	1.0

(N) (Years)	Use Life Probability	Cumulative Probability
6	0.1	0.1
7	0.6	0.7
8	0.3	1.0

Let the Annuity Factor (F) = $((1 + i)^N - 1) / (i(1 + i)^N)$

where N = number of years (useful life)
i = decimal interest rate

and NPV = [(SP - OC) * AS * F] - CC

Using the traditional approach, the NPV would be calculated for the most likely occurrences (AS = 1,200 units, N = 7) and NPV = -$16,000.

Using the rick analysis approach, we let NPV be a random variable because of the uncertainty of AS and N. Many managers are interested in the probability distribution of NPV. We can now estimate the probability distribution of the NPV by a simulation study, by using a random number table or generator to perform the simulation[36] .

An example of assigning random numbers to AS and N is shown below by using a random table.

AS	Probability	Assigned Random Number Table Values
1,000	0.2	00 - 19
1,200	0.5	20 - 69
1,400	0.3	70 - 99

N	Probability	Assigned Random Number Table Values
6	0.1	00 - 09
7	0.6	10 - 69
8	0.3	70 - 99

---

36 Geoffrey Gordon, System Simulation (Second edition. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1978), p. 128.

The user has the option of determining the number of trials, where each trial requires generating 1 random variable for AS and 1 for N. The table below depicts the process for 30 trials.

Trial	Value From Random Number Generator	AS	Value From Random Number Generator	N	F	NPV
1	61	1,200	08	6	3.784	-91,840
2	15	1,000	28	7	4.160	-168,000
•	•	•	•	•	•	•
•	•	•	•	•	•	•
•	•	•	•	•	•	•
9	75	1,400	18	7	4.160	-164,800
30	54	1,200	95	8	4.487	76,880

The user can save the 30 trials for printing later, or request a summary report or frequency diagram of the NPVs. The summary report would have, for example

Probability of a negative NPV = 16/30 trials

positive NPV = 14/30

probability NPV = -168,000 = 5/30

Any variable the user feels is uncertain, the user has the option of determining their probability distributions and generating their values just as above.

3.3.3 Business Planning Program (BPP) Modules

This section describes 4 cases and also discusses other Dynamic Programming Applications. Case 1 uses one Decision Analysis approach for single goals, Case 2 uses Integer Programming, Case 3 uses Goal Programming and Case 4 uses LP, GP, and Priority GP.

BPP Modules Using Decision Analysis, Case 1[37]

Alcott is considering investing in 1 of 3 products, and is interested in one goal-maximizing profit after one year. Profit for these products depends on the status of the economy, which can be strong, flat or weak. PMDSS uses 3 approaches to solve this problem: optimistic, pessimistic, or the decision tree approach. The decision tree approach is depicted in Figure 3.3.3-1. The user inputs the probabilities and PMDSS provides estimates on the profit yield, which also can be changed by the user. The expected value is computed by multiplying the results by their respective probabilities and adding them. The best alternative would be the alternative with the highest expected value.

---

[37] Efraim Turban, Decision Support and Expert Systems: Management Support Systems (Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publisher, London, 1990), pp. 77-79.

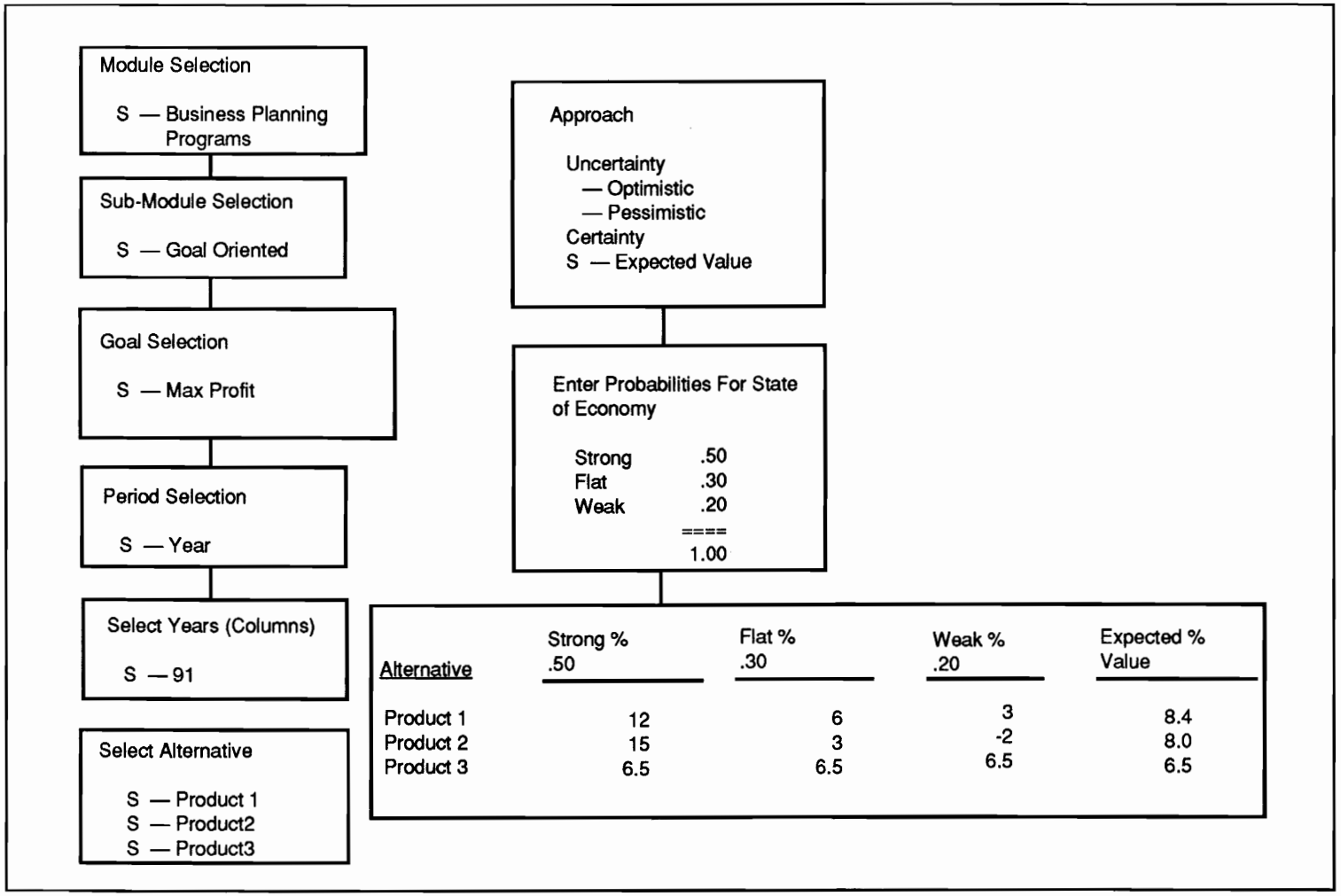


Figure 3.3.3-1. User Interface For Single Goal BPP Model Case 1

BPP Model Using Integer Programming, Case 2[38]

Integer Programming models will be a subset of the BPP Modules. As an example, consider the following situation where the Marketing Department has--options to engage in 6 programs during the next 2 year period. There is, however, only approximately $500,000 available to fund the programs and the manager is interested in knowing which programs should be selected for investment, which will maximize net profit.

The expected costs and expected net profits for the individual programs are listed in the table below:

CASE 2

PROGRAM	EXPECTED NET PROFIT ($000)	EXPECTED COST ($000)
P1	180	125
P2	120	90
P3	100	60
P4	140	120
P5	105	75
P6	200	150

There is $500,000 available for program costs.

Corporate Policy Constraints

1. Exactly 1 of programs P1, P2, P3 must be selected.
2. Exactly 1 of programs P2, P3, P4, P5, P6 must be selected.
3. At most 1 of 2 programs P5, P6, can be selected.

---

38 Thomas W. Knowles, Management Science, Building And Using Models (Irwin, Homewood, Illinois, 1989), p. 487.

4. At most 2 of programs P1, P2, P3, P4, P5 can be selected.

The model for the program selection follows, where $j = 2$ corresponds to program P2, and so forth.

$$\text{Max } Z = 180_{x1} + 120_{x2} + 100_{x3} + 140_{x4} + 105_{x5} + 200_{x6}$$

S.T.

$$125_{x1} + 90_{x2} + 60_{x3} + 120_{x4} + 75_{x5} + 150_{x6} \leq 500{,}000$$

$$x_1 + x_2 + x_3 = 1$$

$$x_2 + x_3 + x_4 + x_5 + x_6 = 1$$

$$x_5 + x_6 \leq 1$$

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq 2$$

with

$x_j = 0$ or 1, $j = 1, \ldots 6$

and

$xj = 1$, if program j is selected
$xj = 0$, otherwise

The first constraint is a budget constraint, while the remaining constraints represent the corporate policy that limits the selection process.

Figure 3.3.3-2 illustrates the User Interface. The selection of programs subject to stated policy is modeled as a 0-1 integer programming problem. In the User Interface, the constraints are formulated by the iterative selection of Logical restrictions. The user selects a contstraint category, (e.g., Mutually Exclusive and Multiple Choice) selects the programs affected (n) and enters (k) presses ENTER to save, returns to the Logical Restrictions Panel (PF15) and repeats the above steps until terminate is selected

from the Logical Restrictions Panel. Terminate triggers a panel to allow the user to enter the total amount available or budgeted, and the output (solution) is displayed in the format the user selected in the beginning of his/her session.

Formulating Integer Programming 0-1 Models[39]

Mutually Exclusive and Multiple Choice Restrictions Case

The jth "yes or no" (select or don't select) decision is represented by the decision variable x_j where

$x_j =$ 1, if decision is yes
$x_j =$ 0, if decision is no.

In general, if there are a set of (n) mutually exclusive decisions where one "yes" decision precludes all others, the condition is represented by the following constraint

$$\sum_{j=1}^{n} x_j \leq 1$$ (at most 1 decision (program) in the set of decisions can be yes)

For example, if at most 1 of 2 programs: (P5, P6) can be selected, then the constraint

$$x_5 + x_6 \leq 1$$

is specified.

On the other hand, if exactly 1 decision out of n decisions must be yes, then the following constraint would be used

$$\sum_{j=1}^{n} x_j = 1$$ (exactly 1 decision in the set the set of decisions must be yes)

---

39 Robert E. Markland And James R. Sweigart, Quantitative Methods: Applications to Managerial Decision Making (John Wiley and Sons, New York, 1987), pp. 502-505.

For example , if exactly 1 of programs P2, P3, P4, P5, P6 must be selected, the following constraint is appropriate

$$x_2 + x_3 + x_4 + x_5 + x_6 = 1$$

A slight extension of this type of constraint involves the situation where k out of n decisions is required to be yes. Two multiple choice constraints for such situations are listed below.

$$\sum_{j=1}^{n} x_j = k \quad \text{(exactly k of n decisions must be yes)}$$

$$\sum_{j=1}^{n} x_j \leq k \quad \text{(at most k of n decisions can be yes)}$$

If, for example, at most 2 of the programs, P1, P2, P3, P4 and P5 can be selected, then

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq 2$$

Precedence or Conditional Relationships

It may be the case that 1 program cannot be selected unless the first program is selected. The logical condition (program k cannot be selected unless program m is selected) is written in the following form

$x_k \leq x_m$ (decision m must be yes for decision k to be yes)

Another conditional relationship might involve the simultaneous selection or non selection of 2 programs. The condition that programs k and m must both be selected would be written as follows:

$X_k = X_m$ (decisions m and k must be the same)

k out of N constraints must be satisfied

In some situations it may be desirable to satisfy many criteria as part of an overall operating policy, although all of these criteria do not hae to be met for the successful operation of the firm. In fact, it may not be possible for all criteria to be met simultaneously. In such situations we require that a given number of conditions (N) be satisfied. If we want to satisfy k out of N conditions, the mathematical structure of this condition would be

$$a_{11}\, x_1 + a_{12}\, x_2 + \ldots + a_{1n}\, x_n \leq b_1 + M y_1$$
$$a_{21}\, x_1 + a_{22}\, x_2 + \ldots + a_{2n}\, x_n \leq b_2 + M y_2$$
$$\bullet$$
$$\bullet$$
$$\bullet$$
$$a_{n1} x_1 + a_{n2} x_2 + \ldots + a_{nn}\, x_n \leq b_n + M y_n$$

with

$$y_i = 0 \text{ or } 1 \qquad i = 1, \ldots N$$

and

$$\sum_{i=1}^{N} y_i = N - k$$

in the above set of conditions M is an arbitrarily large positive number, where the original set of constraints from which a subset is being selected is

$$a_{i1}\, x_1 + a_{i2} x_2 + \ldots a_{im} x_n \leq b_i, \; i = 1, \ldots, N$$

As an example,

$y_i =$ 1, does not require constraint i be satisfied

y_i = 0, requires constraint i be satisfied

and if we consider the first constraint that incorporates the 0 - 1 variable y_i

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 + My_1$$

if $y_1 = 0$ then the above equation becomes

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 + 0$$

if $y_1 = 1$ then the original equation becomes

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq M$$

therefore, the condition

$$\sum_{i=1}^{N} y_i = N - k$$

implies that exactly k of the variables y_i be zero. This in turn means that exactly k of the N conditions must be

Either /Or Conditions

In other situations a choice must be made between 2 constraints, so that 1, but not both, constraints must hold. Consider the following: At least $5000 must be invested in P2, if any money is invested in P2. Let x_j represent the amount of money invested in program j.

Thus we must satisfy

1. $x_j = 0$ or
2. $x_j \geq 5000$

Assuming the non-negativity conditions hold ($x_j \geq 0$) then the above equations become

1. $x_j \leq 0$ or

2. $5000 - x_j \leq 0$

and can be rewritten as

1. $x_j \leq M_y$ or
2. $5000 - x_j \leq M(1-y)$ where $y = 0$ or 1

Here the 0 - 1 variable was incorporated in a manner similar to the previous example from the "k out of N Constraints" section.

We note that if $y = 0$, x_j must be equal to zero, and if $y = 1$, x_j must be greater than or equal to 5000.

Fixed Charge Condition

Many applications involve situations where a fixed-charge or setup cost is experienced if a particular activity (program) is undertaken. The total cost of the program must reflect both the fixed charge for this activity and the variable cost of the activity. If x_j represents the activity level, c_j the per-unit activity cost, and k_j the fixed charge then:

if $x_j = 0$, total cost $= 0$

if $x_j > 0$, total cost $= k_j + c_j x_j$

Using a 0-1 variable to impose the above conditional relationships can be done by requiring

$x_j - My_j \leq 0$

total cost $= k_j y_j + c_j x_j$

where y_j is a 0 -1 variable.

When $x_j > o$, $y_j = 1$ so that

$x_j \leq M$ and

total cost $= k_j(1) + c_j x_j$

When $x_j = 0$, $y_j = 0, 1$. Since the objective in this case would be to minimize cost, $y_j = 0$ and total cost $= 0$.

Given this procedure for handling a fixed--charge situation, the general fixed charge model for n activities.

$$\text{Min } Z = \sum_{j=1}^{n} (c_j x_j + k_j y_j)$$

$$\text{S.T.} \quad x_j - My_j \leq 0$$

with

$$x_j \geq 0 \quad j = 1, \ldots, n$$

and

$$y_j = 0 \text{ or } 1, \qquad j = 1 \ldots, n$$

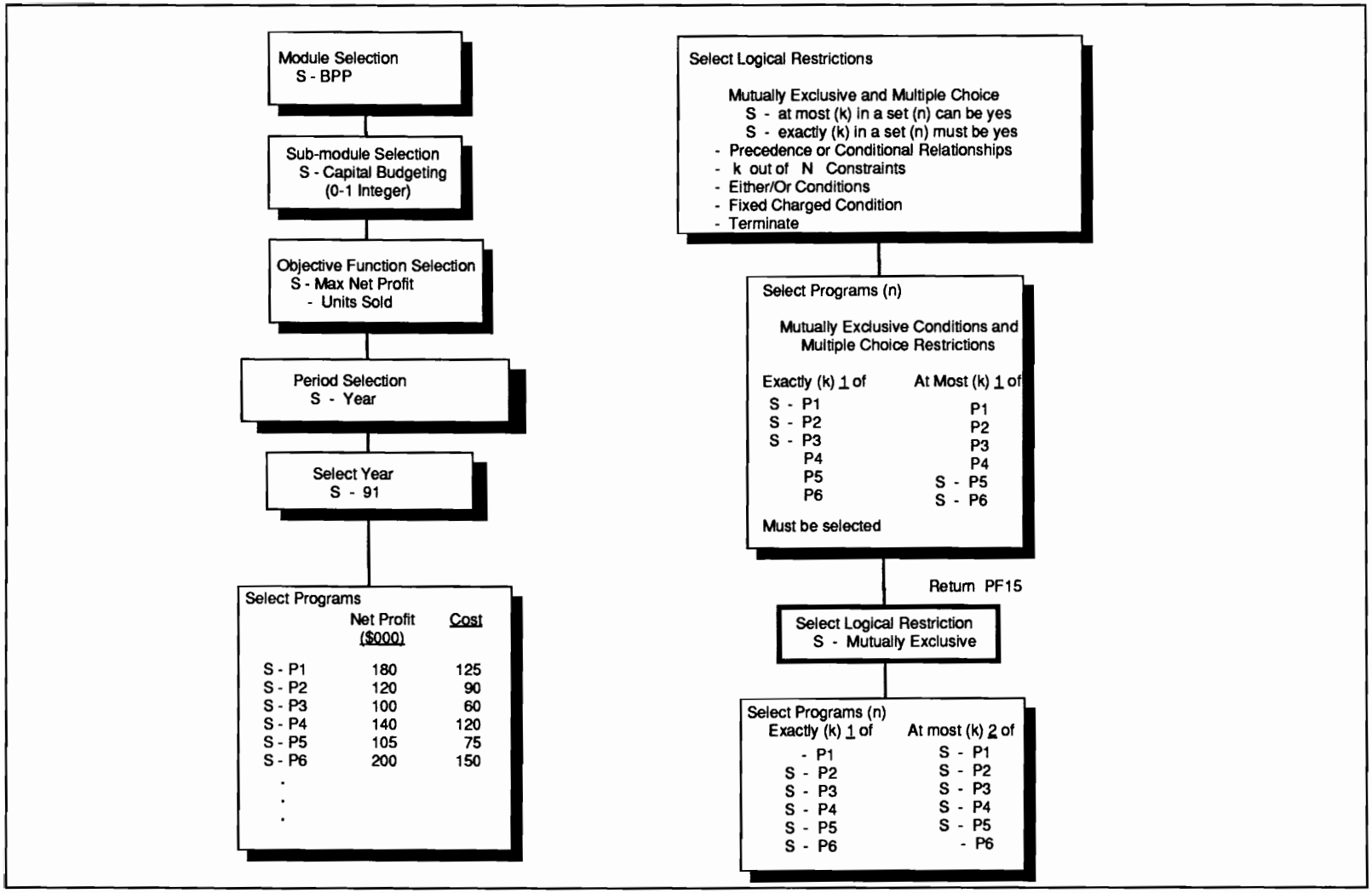


Figure 3.3.3-2. User Interface For BPP Interger Programming Model Case 2

BPP Model using Goal Programming, Case 3[40]

As another example, suppose Alcott is planning another advertising program for the coming year to promote a new product (Suntan Lotion). The cost per ad placed in each of the media and their reach (in terms of person-exposures) are given in the table below.

MEDIA	COST/AD	FEMALES	MALES
1	$15	70	40
2	11	40	25
3	18	60	100

For example, each ad in medium 1 costs $15 and reaches 110 people, 70 of whom are females.

Alcott's program manager wants the following items considered in formulating a Goal Programming (GP)

- its advertising budget is $300 weekly
- goal 1 is to reach 2,500 person per week
- goal 2 is to minimize the under achievement of reaching 2,000 females/week

Females are a desirable target market for Alcott's new Suntan Lotion.

- goal 3 is to minimize the amount of money spent in excess of $250 per week on media 2 and 3 combined.

---

[40] Michael Q. Anderson And R. J. Lievano, Quantitative Management: An Introduction (Second edition. Kent Publishing Co., Boston, Massachusetts, 1986), pp. 520-521.

- goal 1 is 3 times as important as goal 3 and 1.5 times as important as goal 2.

The User Interface has to incorporate all possible combinations to accomodate user preferences. The total number of possible combinations will be determined after the user has made her media selection (i.e., if a user chooses Media 1, 2, and 3 then x_1, x_2, x_3 would represent the decision variables for the number of ads placed per week in each respective media). Interactively, the user will then specify goals (select) and specify the relative importance (goal 1 is 3 times as important as goal 3).

A GP can then be formulated to determine a satisfactory media mix.

Let $x_1, x_2\ x_3$ ~ number of ads placed per week in media 1, 2, 3

d_1+, d_2+, d_3+ ~ overachievement of goals 1, 2, 3

d_1-, d_2-, d_3- ~ underachievement of goals 1,2, 3

$$\text{Min } Z = 3d_1^+ + 3d_1^- + 2d_2^- + d_3^+ \qquad \text{(relative importance)}$$

$$\begin{aligned} \text{S.T.} \quad & 15x_1 + 11x_2 + 18x_3 && \leq 300 \\ & 110x_1 + 65x_2 + 160x_3 - d_1^+ + d_1^- && = 2{,}500 \\ & 70x_1 + 40x_2 + 60x_3 - d_2^+ + d_2^- && = 2{,}000 \\ & 11x_2 + 18x_3 - d_3^+ + d_3^- && = 200 \end{aligned}$$

with

$$x_1, x_2, x_3, d_i^+, d_i^- \ (i = 1,2,3) \qquad \geq 0$$

The solution can be obtained by using a canned LP (simplex) program.

BPP Models Using L P->GP->Priority GP (Approaches to multiple objectives) Case 4[41]

The business planning models will use 3 approaches for solving problems with multiple objectives. The 3 approaches will be illustrated using the following data:

Media	($000) Cost/Ad	(000) Total Exposures	(000) High-Income Exposure
1	10	18	1
2	9	15	1.1
3	1.5	1.8	.120
4	2.5	1.9	.5

The objective here is to minimize the total cost of the advertising program:

1) # of dollars spend ≤ 60

2) # of Total exposures ≥ 110

3) # of high income exposure ≥ 7.5

4) # of media 3 ads ≥ 4

Let

x_1 ~ number of Ads in media 1

x_2 ~ number of Ads in media 2

x_3 ~ number of Ads in media 3

x_4 ~ number of Ads in media 4

then the model becomes (Approach 1, LP)

$$\min \quad 10x_1 + 9x_2 + 1.5x_3 + 2.5x_4$$

$$\text{S.T.} \quad 10x_1 + 9x_2 + 1.5x_3 + 2.5x_4 \leq 60$$

$$18x_1 + 15x_2 + 1.8x_3 + 1.9x_4 \geq 110$$

$$x_1 + 1.1x_2 + .120x_3 + .5x_4 \geq 7.5$$

---

41 Thomas W. Knowles, Management Science, Building And Using Models (Irwin, Homewood, Illinois, 1989), pp. 320-328.

$$x_3 \geq 4$$

with

$$x_1, \quad x_2, \quad x_3, \quad x_4 \geq 0$$

The user encounters an infeasible solution, and must choose the goal oriented (GP) approach 2. Lets say requesting 2 goals.

goal 1 ~ come close as possible to the total exposure goal

goal 2 ~ come close as possible to high income exposure goal

high-income is 7 times as important as total exposure

Let d_1^+, d_2^+ ~ overachievement of goals 1, 2

d_1^-, d_2^- ~ underachievement of goals 1, 2

Now the model becomes

$$\min \quad d_1^- + 7d_2^-$$

$$\begin{aligned}
\text{S.T.} \quad & 10x_1 + 9x_2 + 1.5x_3 + 2.5x_4 && \leq 60 \\
& 18x_1 + 15x_2 + 1.8x_3 + 1.9x_4 - d_1^+ + d_1^- && = 110 \\
& x_1 + 1.1x_2 + .12x_3 + .5x_4 - d_2^+ + d_2^- && = 7.5 \\
& x3 && \geq 4
\end{aligned}$$

with

$$x_1, \quad x_2, \quad x_3, \quad d_i^+, \quad d_i^- \ (i = 1, 2) \geq 0$$

The goal programming formulation assigns weights to the amounts that goals were undersatisfied or underachieved. This approach can be used for as many objectives as desired.

Another approach to multiple objectives is priorities, where you establish a hierarchy of goal priorities. Using priority goal programming, (Approach 3) you

optimize the first goal and then, using the second goal, you choose from among the solutions that optimize the 1st goal. Let's say the priority 1 goal is total exposures and priority 2 goal is # of high income exposures.

Because the goals conflict, you must either solve the problem in 2 stages using LP or use priority goal programming software.

The stage 1 model using LP becomes

$$\min \; d_1^-$$

$$\begin{aligned}
\text{S.T.} \quad & 10x_1 + 9x_2 + 1.5x_3 + 2.5x_4 && \leq 60 \\
& 18x_1 + 15x_2 + 1.8x_3 + 1.9x_4 - d_1^+ + d_1^- && = 110 \\
& x_1 + 1.1x_2 + .12x_3 + .5x_4 - d_2^+ + d_2^- && = 7.5 \\
& x_3 && \geq 4
\end{aligned}$$

with

$$x_1, \quad x_2, \quad x_3, \quad x_4 \qquad \geq 0$$

Let k = value of stage 1 optimal objective function then the Stage 2 Model becomes

$$\min \; d_2^-$$

$$\begin{aligned}
\text{S.T.} \quad & 10x_1 + 9x_2 + 1.5x_3 + 2.5x_4 && \leq 60 \\
& 18x_1 + 15x_2 + 1.8x_3 + 1.9x_4 && = 110 + k \\
& x_1 + 1.1x_2 + .12x_3 + .5x_4 - d_2^+ + d_2^- && = 7.5 \\
& x_3 && \geq 4
\end{aligned}$$

with

$$x_1, \quad x_2, \quad x_3, \quad x_4 \qquad \geq 0$$

Dynamic Programming Applications

The objective in dynamic programming is to allocate the limited resources to the activities (e.g., programs) in order to optimize an overall measure of effectiveness[42] . Such resource allocation problems can be incorporated throughout the other modules in the DSS. A list of such problems is presented below:

BPP Marketing Module

a. How should an advertising budget be allocated to several media to maximize the number of personnel exposed to a product in a given time span, where the limited resource is the amount of money in the budget? (Advertising Effectiveness)[43]

b. Suppose Alcott has in units of product to distribute to M markets to be sold. The sales potential of the M markets differ from each other. How should Alcott distribute (allocate) its product to the markets in order to maximize total profit of n units? The limited resource in the finite quantity of n items to be sold (product distribution).

---

[42] Richard E. Trueman, Quantitative Methods for Decision Making in Business (Dryden Press, 1981), p. 439.

[43] Robert J. Thierauf And Robert C. Klekamp, Decision Making Through Operations Research (Second edition. John Wiley and Sons, Inc., New York, 1975), pp. 512-513.

Other R&D Applications[44]

a. Suppose there are N people available for assignment to M programs. (e.g., assign Technical personnel to research projects) How many people should be assigned to each program (project) in order to maximize the total profit from the projects? The limited resource is the group of people available for assignment. The return from each project depends on how many people are assigned to it. This return may be an expected return (in a probabilistic sense).

---

[44] James R. Burns And Larry M. Austin, Management Science Models and the Microcomputer (Macmillan Publishing Company, New York, 1985), p. 152.

3.3.4 Manpower Planning Program (MPP) Modules

This section discusses 3 cases using Dynamic Programming. Case 1 solves an allocation problem: Case 2 discusses allocation problems using probabilistic DP; Case 3 solves a scheduling problem.

MPP Model using Dynamic Programming, Case 1[45]

The Manpower Planning Program Modules will be used to assist the program Managers in allocating manpower resources and for manpower scheduling. A typical example would be the allocation of salesmen by region, product or any of the predefined dimensions from the User Interface. Dynamic programming can be used to solve an allocation problem of this type.

As an example consider the following: One of Alcott's program managers has enough money in his budget to hire 5 salesment o cover 3 regions. He wishes to determine the most profitable allocation, based on estimates of monthly profit as a function of the number of salesmen assigned to each region.

Formulating this problem as a multistage decision process just as in the R&D example, the following variable definitions apply:

n = Total number of regions (stage or point in problem where a decision must be made)

T = Total number of salesmen available for assignment

s_i = Total number of unassigned salesmen at stage i $(0 \leq s_i \leq T)$

---

[45] Richard E. Trueman, Quantitative Methods for Decision Making in Business (Dryden Press, 1981), pp. 413-419.

A_i = number of salesmen to be assigned to region $_i$ at stage $_i$ (resource to be allocated) $(0 \leq A_i \leq T)$

$R_i(A_i)$ = immediate return function (return from assigning s_i salesmen to Region i)

$f_i^*(s_i)$ = optimal return function (maximum return when s_i salesmen are allocated in an optimal fashion at stage $_i$ and all preceding stages)

The objective in this problem is to maximize the total profit subject to the limitation on the number of salesmen available. We can state this as follows:

$$\text{maximize} \quad \sum_{i=1}^{n} R_i(A_i)$$

$$\text{S.T.} \quad \sum_{i=1}^{n} A_i \leq T$$

$$\text{where } A_i = 0, 1, \ldots, 5 \text{ for all } i$$

A typical User Interface session is depicted in Figure 3.3.4-1.ß

As in the Research and Development example, we have to establish a basic recursion relationship, whereby the return at each stage is expressed as a function of data relevant to that stage and the optimal return function of the previous stage.

If the user wishes to evaluate the optimal assignments fora smaller number of salesmen, (e.g., 4 salesman were available) we could immediately go to stage 3, with s_3 = 4, and utilize the optimal assignments perviously determined at stage 2. If the program manager wishes to determine the optimal allocation of salesmen when the number available ranges from 1 to 5, the original return functions (estimated profits)can be used to recalculate the optimal allocations. This type of information could be quite useful to management, which might be in the position of determining how many salesmen to allocate to the 3 regions combined. Thus, the problem of optimal allocation

of the salesmen to these 3 regions could be a subproblem of a much larger optimization problem.

MPP, Case 2[46]

As another example relating to manpower allocations, suppose the returns from assigning a certain number of salesmen to each region were represented by probability distributions, rather than single values. In this situation probabilistic distributions add little, if any additional complexity because the random elements in the problem do not affect the actual state transformations. If we wished to optimize in terms of expected monetary value, the optimal solution would be determined by placing each probability distribution by its mean value and solving exactly as in the deterministic case for the manpower planning example.

Manpower Planning Programs (Scheduling), Case 3[47]

Consider the situation where Alcott has manpower commitments to a certain program as a project. Each period (month) a certain required minimum manpower level has been determined. Changing this level each period results in employment or layoff costs, depending on the direction of the change. If the manpower level is higher than required in any month, the company is paying for idle manpower. Over the time span of the program, the company wishes to determine the least-cost manpower schedule.

As a simple version of this problem, suppose that the cost of changing manpower level is proportional to the square of the monthly change in manpower level

---

46 Ibid., p. 432.

47 Ibid., pp. 423-430.

(either up or down) and that the cost of excess personnel is a linear function of the number of such personnel. Each period's minimum manpower requirement must be met.

Let M_i = minimum manpower requirement in period i

n_i = manpower assigned in period i

C_1 = cost factor for a change in monthly manpower between periods

C_2 = cost for each excess person per period

For month i, the cost of changing the manpower level is $C_1 (n_i - n_{i-1})^2$, and the cost of excess personnel is $C_2(n_i - M_i)$, where $n_i \geq M_i$. If there are N periods and the manpower level n_0 is known prior to the start of the project, then our objective is to:

$$\min \quad \sum_{i=1}^{N} [\, C_1(n_i - n_{i-1})^2 + C_2(n_1 - M_i)]$$

$$\text{S.T.} \quad n_i \geq M_i, \quad i = 1,2, \ldots . N$$

To formulate as a dynamic programming problem, if let each stage represents a time period, then

S_i = manpower level at the beginning of period i (state at stage i)

D_i = manpower level during period i (decision at stage i)

stage 1 = last period

The manpower at the beginning of any period must be the manpower assigned in the previous period, so that

- state S_i at period i $= n_{i+1}$ (manpower assigned in period i+1, the immediately proceding period, using the backward numbering scheme)

At the period N, with state $S_i=n_2$, we must minimize the sum of the 2 cost components for each feasible manpower level n, so the optimal return function is

$$f_1(n_2) = \min_{n_1 \geq M_1} [C_1(n_2 - n_1)^2 + C_2(n_1 - M_1)]$$

For any period earlier than period N, the lowest achievable cost associated with a given manpower level n_i at that stage is calculated by summing the cost of changing from manpower level n_{i+1} to N_i, the execess manpower cost for n_i, and the minimum cost for all lowered numbered stages (later periods) when starting with manpower level n_i at this stage.

Let $f_i(n_{i+1})$ = minimum cost when manpower level n_{i+1} exists at the beginning of period i and an optimal manpower allocation is made to stage i and all lower numbered stages

By definition, the minimum cost for all lower number stages is then $f_{i-1}(n_i)$ when manpower level n_i is chosen. The general optimal return function can now be expressed as

$$f_i(n_{i+1}) = \min [C_1(n_{i+1} - n_i)^2 + C_2(n_i - M_i) + f_{i-1}(n_i)], \qquad 1 \leq i \leq N$$

.

In general dynamic programming attempts to reduce a single problem involving a relatively large number of decisions variables, frequently integer-values, into a series of smaller problems, each with a small number of decision variables. By so doing, it avoids the computational problem of evaluating all possible combinations of variables. Most of these combinations need not to be evaluated if some reasonable weeding-out process is performed. However, if there are more than a very few state variables

(perhaps 2) at each stage, unless they have a very small number of possible states, the number of required values to be retained in computer memory rapidly, become excessive. Thus, if we had 3 state variables, each with 100 possible states, we would have to retain $(100)^3$ or 1,000,000 values in computer memory at one time. There are, in practice, several different methods by which the number of state variables can be reduced, where the best known method of state variable reduction is through the use of Lagrange multipliers.

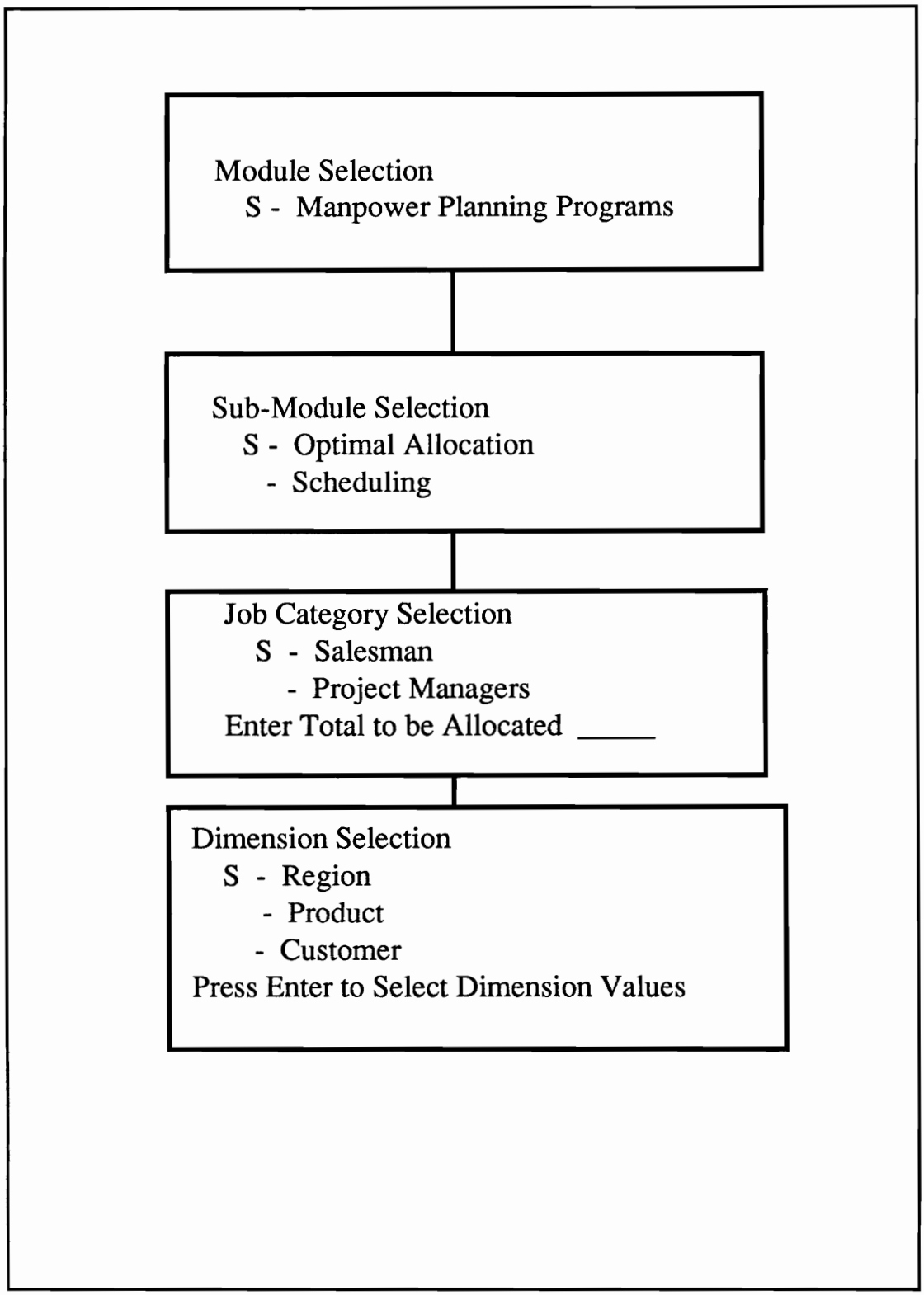


Figure 3.3.4-1. User Interface For MPP Dynamic Programming Model Case 1

3.3.5 Executive Support Module

Executive Support Module consists of several submodules:

- Profs Interface
- Dow Jones Interface
- Nielson Market Analysis Interface

IBM VM Profs assists the user in administrative functions as:[48]

- Calendar - where a user enters his schedule (meetings, appointments, actions due) and views other user schedules
- Telephone and address lists
- Mail - where the user sends and receives mail or memos
- Reminder provides a flash message to remind the user of important entries from calendar

The Dow Jones interface provides access to outside news and company information[49] . The Dow Jones News (Retrieval Services can assist the executive team in strategic decision making by giving them information about government, business and economic conditions, including Alcott's stock, competitor's stock, other current or historical stock quotes. The news is also categorized by time frame (current), industry and company. Dow Jones News and stock information will give the executive team an edge in monitoring acquisition candidates and tracking marketing trends.

48Efraim Turban, Decision Support and Expert Systems: Management Support Systems (Second Edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990), p. 713.

49 Ibid., p. 241.

The Nielson Market Analysis provides market survey data in a relational database. PMDSS will provide a set of canned queries to interrogate the database by[50] :

- demographics
- product
- product category

This data should assist the Business Planning PM in the following manner:

- Target products where they are weakest
- Test new products in experimental areas
- Identify new markets for existing products
- Identify new products / product categories (product trends, product demand)
- Examine consumer behavior
- Compare strategies (other competitors)

---

[50] Interview with William White, IBM, Dallas, Texas, February 1990.

3.4 PMDSS Database

3.4.1 Sources of Input

We have identified sales as a critical success factor for Alcott, as well as demonstrated how The Business Planning Programs play a key role in the success of Alcott. Alcott markets its products in the United States as well as abroad. Each sales region is serviced by several sales people; the sales people are managed by sales managers, sales managers are managed by project managers, where usually there is a 1-1 relationship between product (line) category and project managers; project managers are in turn managed by program managers.

Alcott's MVs mainframe computer handles all order processing, inventory control, personnel and accounting functions. The MVS mainframe computer also runs other decision support models, such as projected sales by various time periods and extracts the outputs in theVM mainframe. Stored within the mainframes are files and databases that track sales and projected sales and other data needed to support the executive teams' requirements. Extracts of operational data as well as model data must be periodically (daily, monthly) downloaded (copied) from the mainframe ultimately to the PC workstation.

Program managers, as part of their duties, develop an annual product plan for each major product to establish, among other things, the sales goals for the product and the budget for marketing. Marketing expenses include:

- advertising
- salesperson training
- dealer promotion

- etc.

To assist the program manager, PMDSS needs to know, for example, product sales by region, the effect some aspect of marketing-say, advertising-has had a product sales, what was the total sales figure for a certain product in a particular month, how does this compare to a previous month, how can I improve sales for next quarter.

The user interface was emphasized in the study to identify the various objects that Program Managers need. From objects, a relational database design can be derived [51] and the database management system can be implemented.

3.4.2 Derivation of PMDSS Database

The suggested relational DBMS for this study is Dbase IV. Dbase IV uses the relational data language SQL. Dbase IV will be used to manage the queries, security, retrieval and changes of the PMDSS data, but the PMDSS's user perception will largely be based on the performance (timing) of this management.

One objective in PMDSS is to provide a "good" relational database design. This task is most critical. This section describes the following:

- relational model terminology
- relational database design
- data extraction design

---

[51] Sally Shaer And Stephen J. Mellor, Object-Oriented Systems Analysis: Modeling the World in Data (Yourdon Press, Prentice Hall Building Englewood Cliffs, New York, 1988), p. 41.

Relational Model Terminology

The relational model is a way of looking at data and is concerned with 3 aspects of data: data structure, data integrity and data manipulation.

The Relational Model Components As Defined By Codd are[52] :

Data Structure

- domains (values)
- N-ary relations (attributes, tuples)
- keys (primary, foreign)

Data Integrity

- primary key values must not be null
- foreign key values must match primary key values

Data Manipulation

- relational algebra
  union, intersection, difference, product, select, project, join, divide
- relational assignment

A relation is a two-dimensional table that has single-value entries. Entries in a given column are all of the same kind, columns have a unique name and the order of columns is immaterial. Columns are called attributes. No two rows of a table are identical and the order of rows in the table is also immaterial[53] . Rows are called tuples. Sometimes the term table or file are substituted for relation, the terms column or field are substituted for attribute, and the terms row or record are substituted for tuple. A

---

52 C. J. Wate And Colin J. White, A Guide to SQL/DS (Addison-Wesley Publishing Company 1989), p. 451.

53 Rex Hogan, A Practical Guide To Data Base Design (Prentice Hall, Englewood Cliffs, New Jersey, 1990), p. 104.

relational model limits the data structures of the table. A domain is a set of values, all of the same type[54] . A primary key is a unique identifier[55] for each row in a table. Columns that contain the same data values in other tables are foreign keys. [56]

Based on the User Interface for the Business Planning Programs, suppose the following relations were developed (an underlined column represents a primary key). These relations will be used for examples throughout this section:

INVOICE	(product-number, invoice-date, total, salesperson- employee-number, customer-number, customer-name)
LINE-ITEM	(invoice-number, line-item-number, product-number, quantity . . .)
CUSTOMER	(customer-number, customer-name, . . .)
SALESPERSON	(salesperson-employee-number, name, region . . .)
PRODUCT	(category, product-number, name, product-price . . . .)
MEDIA-AD	(media, date, media-name, cost . . . .)
PRODUCT-MEDIA	(media-name, product-number)
PRODUCT-SALES	(invoice-date, product-number, product-price)

---

54 Ibid.

55 Ibid.

56 Ibid.

CUST-INVOICE (customer-number, invoice-number)

Data Manipulation Relational Algebra

The manipulative part of the relational model consists of a set of operators known as relational algebra. Each operator of the relational algebra takes either one or two relations as its operand(s) and produces a new relation as its result[57] . SQL constructs are simple. The select, from and where clauses are flexible and will be used to illustrate Codd's ( ) originally defined 8 operators, 2 groups of 4 each: (1) The traditional set operations (union, intersection, difference and Cartesion product) and (2) special relational operations (select, project, join and divide)[58] .

Traditional Set Operations

- Union

  Select invoice-number from **INVOICE**
  Union
  Select invoice-number from **LINE-ITEM**

- Intersection

  Select invoice-number from **INVOICE**
  where exists
  (select invoice-number from **LINE-ITEM**
  where **LINE-ITEM**.invoice-number = **INVOICE**.invoice-number

- difference (combines 2 tables to produce a 3rd table that contains all rows that are in the **LINE-ITEM** table but not in the **INVOICE** table

---

[57] C. J. Date, A Guide To DB2 (Addison-Wesley Publishing Company, 1984), p. 270.
[58] Ibid.

Select invoice-number from invoice
where not exist
(select invoice-number from **LINE-ITEM**
where **LINE-ITEM** . invoice-number = **INVOICE** invoice-number

- product (concatenate each row of **LINE-ITEM** with each row of **INVOICE**)
  Select **INVOICE.***, **LINE-ITEM** .* from **INVOICE-LINE-ITEM**

Special Relational Operations

- selection (allows a table to be formed that contains all columns from selected rows in original table.)
  Select *
  From **PRODUCT**
  where category = '5'

- projection (yields a "vertical" subset of a given relation)
  Select distinct name
  from **CUSTOMER**

- join (combines rows from 2 tables, resulting in a 3rd, wider table where the join operation is based on 1 or more columns from each of 2 tables whose data values share a common domain)

  Select **INVOICE.*, SALESPERSON.***
  From **INVOICE, SALESPERSON**
  Where **INVOICE**.salesperson-employee-number=**SALESPERSON**.salesperson-number

- divide - divides a relaton of degree 2 (dividend) by a relation of degree 1 (divisor) and produces a result relation of degree 1 (quotient)

  Select distinct customer-number from **CUST-INVOICE, CUSTOMER**
  where not exists

```
( select invoice-date from PRODUCT-SALES
where not exist
   (select  * from CUST-INVOICE, INVOICE
      where INVOICE.customer-number=CUSTOMER.customer-
            number
      and  INVOICE.invoice-date = PRODUCT-SALES.invoice-
            date))
```

Here we are assuming (a) relation **PRODUCT-SALE** has 1 attribute. We divide the 1st relation (**CUST-INVOICE**) by the 2nd relaton (**PRODUCT-SALES**) and obtain a result (customer-number)[59]

Relational Assignment

Relational assignment operations allow the value of some algebra expression--say a join-to be saved in the database. This can be simulated in SQL with the insert...selection operaton.

Relational Database Design[60]

The logical steps involved in designing a relational database are:

- decide what data relationships to record in the database.

  Example

  products are classified by category in relation **PRODUCT**

- define tables for each type of relationship

  -- one-to-many

---

59 James Martin, DB2: Concepts, Design, and Programming (Prentice Hall, Englewood Cliffs, New Jersey, 1989), pp. 17-31.

60 IBM, SQL/Data System: Database Planning and Administration for VM/System Product and VM/Extended Architecture System Product (Version 2, Release 2, 1988), pp. 1-22.

-- many-to-one

-- many-to-many

-- one-to-one

- provide column definitions for all tables
- identify one or more columns as a primary key

  Example

  product-number is the primary key for relation **PRODUCT**

- be sure that equal values represent the same entity

  Example

  invoice-number in **INVOICE**=invoice-number in **LINE-ITEM**

- plan for referential integrity (all references from 1 table to another are valid)
- normalize your tables (normalization is a method to organize your data into tables that contain the least amount of redundant information). The goal of normalization is to arrange all data in simple tables. In each table, each row consists of:

  -- a key that uniquely identifies some entity

  -- a set of mutually independent values that describe the entity

  Normalization is a sound, technique for data structure design because it increase data independence and logical data integrity but it is not required of Dbase IV.

Data Extraction Design

The PMDSS must represent its data at various levels of details and dimensions, with varying amounts of data accomodating varying time frames from multiple sources. This is a big problem and must be considered carefully in the analysis and design and

implementation activities of the project life-cycle. As an example, consider some of the data extract routine queries PMDSS must use to aggregate and subset[61] : Suppose APMDS needs to know which customer purchased a product category in March. The following SQL query would extract the data.

```
Select distinct customer-name from CUSTOMER
Where customer-number in
        Select customer-number from INVOICE
        Where date between 900301 and 900331
                Select invoice-number from LINE-ITEM
                Where LINE-ITEM.product-number in
                        Select product-number from PRODUCT
                        Where category = '5'
```

Interpreting this example from the bottom-up, we begin by building a list of product category = '5', build a list of invoice numbers that line-items numbers for all those invoices occuring in March and build a list of customer names, eliminating duplicates. As another example, suppose PMDSS needed the products that were advertised in March. The following SQL query would extract the data:

```
Select product-number, name from PRODUCT
Where product.number in
        Select distinct product-number from PRODUCT-MEDIA
        Where PRODUCT-MEDIA.media-name = MEDIA-AD.media-name
        And MEDIA-AD.date between 900301 and 900331
```

---

[61] Ralph H. Sprague, Jr. And Eric D. Carlson, Building Effective Decision Support Systems (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982), p. 247.

This example extracts data from several tables. Product name comes from **PRODUCT** table. Advertisements are stored in **MEDIA-AD**. Advertisements are not directly associated with any product, but there associations can be derived and similarly, products are not directly associated with an advertisement. As a final example, consider PMDSS needs the total sales for product '123' for the month of March. The following SQL query would extract the data.

Select sum (product-price) from **PRODUCT-SALES**
Where invoice-date between 900301 and 900331
And **PRODUCT-SALES**-number = '123'

In this example, a subset of the larger data has been established in **PRODUCT-SALES**. SQL's built-in function sum calculates the totals[62].

---

[62] David M. Kroenke And Katheleen A. Dolan, Database Processing: Fundamentals Design And Implementation (Third edition. Science Research Associates, Inc., 1988), p. 425.

CHAPTER 4

PROJECT CHARTER

4.1 Developmental Approach

Two approaches are considered in managing the PMDSS system life cycle:

- structured project life cycle
- prototyping life cycle

Structured Project Life Cycle

The structured project life cycle is depicted in Figure 4.1-1 with 9 activities and 3 terminators. The rectangular boxes (terminators) are the ultimate recipients of the system. These 3 entries interact with the 9 activities shown:[63]

1. Study. This study started when the CIO made her request. Its purpose was to identify the current deficiencies in the executive team's environment, establish new goals, determine whether it is feasible to automate some executive functions, suggest some acceptable scenarios, and propose system summary.
2. Analysis. The primary purpose of the analysis activity will be to transform the proposed system into a structured specification (essential model), which involves modeling the executive team's environment with data flow diagrams and other graphic documentation tool.
3. Design. The activity of design will allocate portions of the essential model to processors (CPUs) and to appropriate tasks (jobs) within each

---

[63] Edward Yourdon, Managing The System Life Cycle (Second edition. A Prentice Hall Co., Englewood Cliffs, New Jersey, 1988), p. 27.

processor. Within each task, the hierarchy of program modules and their interfaces are developed.

4. Implementation. This activity includes both coding and the integration of modules into a more complete skeleton of the ultimate system, using both structured programming and top-down implementation.[64]
5. Acceptance Test Generation. Since the structured system will define an acceptable system from the user's point of view, work can commence on the activity of generating a set of acceptance test cases.
6. Quality Assurance. This activity requires, as input, acceptance test data generated in activity 5 and an integrated system produced by activity 4. Quality assurance is also known as final testing or acceptance testing.
7. Procedure Description. The output of this activity is a user's manual.
8. Database Conversion. This activity will involve more work and perhaps more strategic planning than the development of programs within the modules. This activity is most critical, and probably will be 1 of the determining factors of the project's success or failure.
9. Installation. Its inputs are the user's manual, the PMDSS data base and the accepted system.

The data flow diagram illustrated does not imply that all of the N activities must finish before activity N+1 commences, but strangely implies that several activities may be going on in parallel. This is the reason for using the work "activity" rather than phase. The term phase is used in a Systems engineering context. Figure 4.1-2 depicts the System Life Cycle as it relates to Systems Engineering.

---

64 Ibid., p. 56.

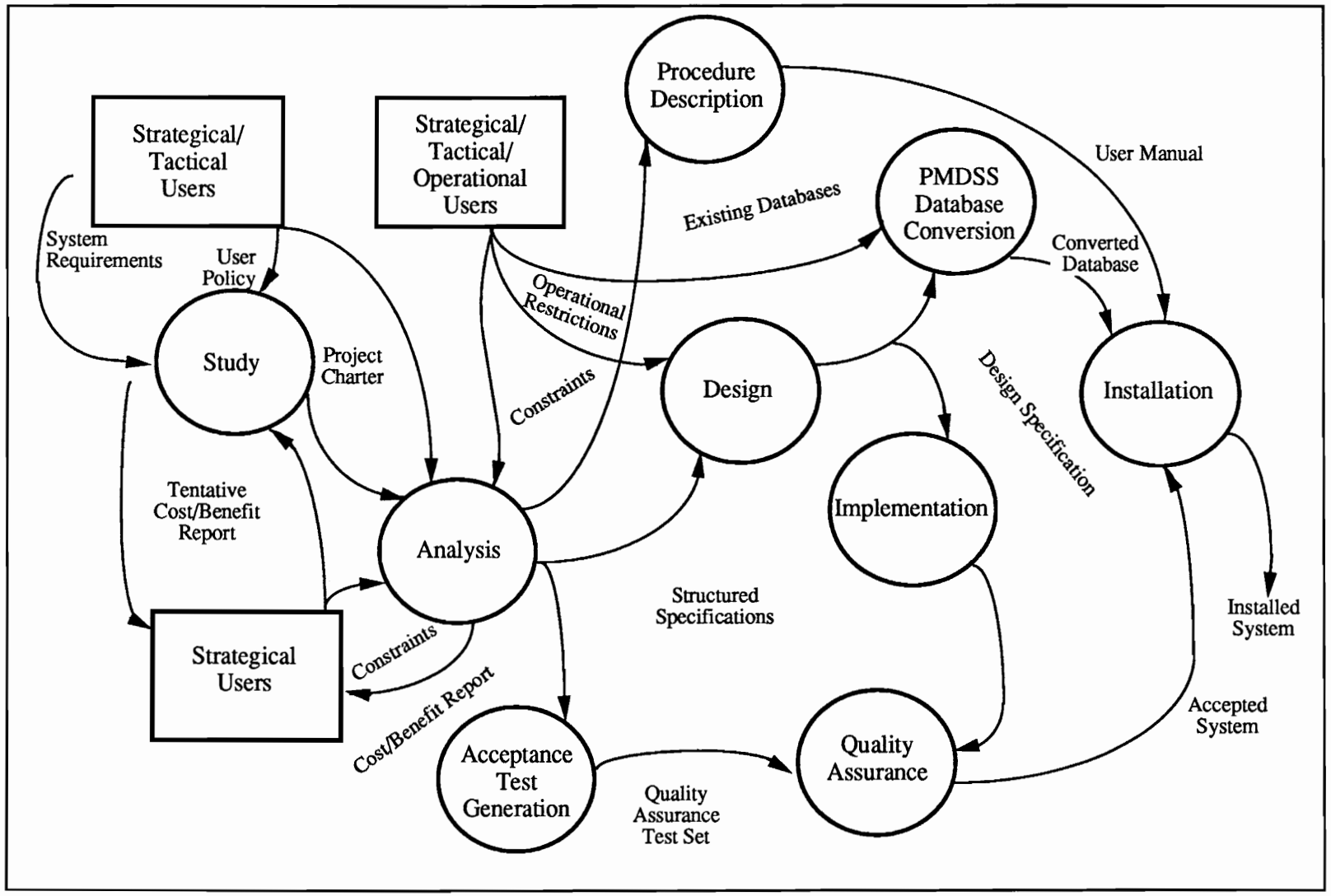


Figure 4.1-1. PMDSS Structured Project Life Cycle

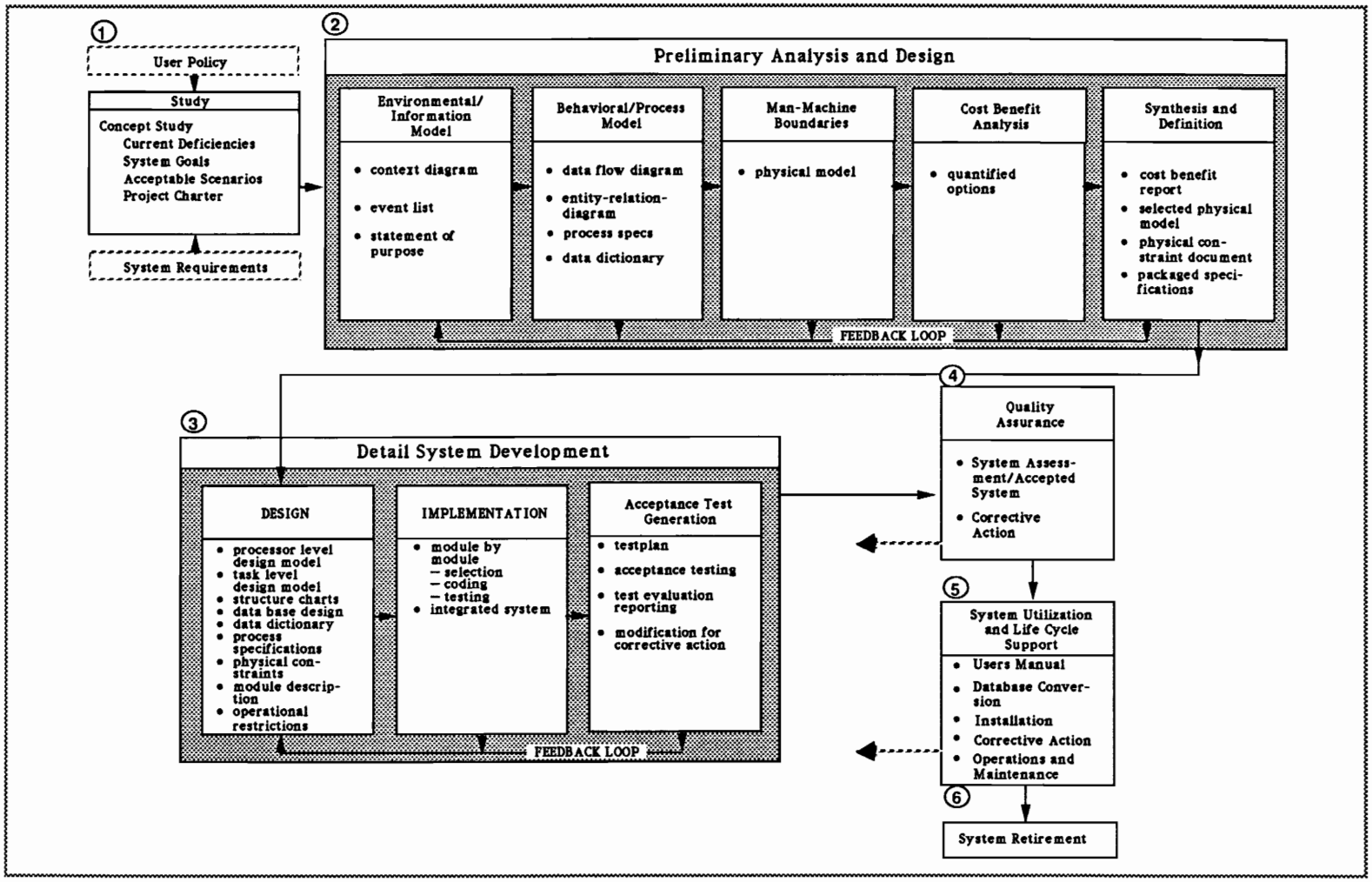


Figure 4.1-2. PMDSS System Life Cycle From A Systems Engineering Point of View
Source: Benjamin S. Blanchard And Wolter J. Fabrycky, Systems Engineering And Analysis (Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981), p. 453.

Also, the data flow diagram does not explicitly show feedback or control. Virtually every activity produces information that can provide suitable modifications to 1 or more of the preceding activities. For example, the design activity could produce information that may revise some of the cost/benefit decisions in the analysis activity, or knowledge gained in the design activity may require revising earlier decisions made in the study, or certain events in any activity could cause the entire project to terminate.

The activities described above are macro in focus. The micro level subactivities for each activity will require careful planning and consideration.

Jordan suggests (based on statistics from other structured project life cycle projects) the following division of labor among the activities.[65]

Study	5%
Analysis	35%
Design	20%
Implementation	15%
Remaining	25%

This study took 2 months with 1 person full time. Also note that 60% of the project resources will be consumed before the first line of code is written. Alcott's strategic users must determine if this is politically tolerable. The user may become disinterested if he sees no tangible results within a 90 day time frame, for example.

---

65 Ibid., p. 148.

Prototyping Life Cycle

The prototyping approach has been popularized by Bernard Boar and James Martin. Bernard Boar describes it in Application Prototyping (1984) as system modeling or heuristic development and suggests it offers an attractive and workable alternative to prespecification methods to deal better with uncertainty ambiguity and fickleness of real-world projects.[66]

The primary difference between the structured approach and the prototype approach is:[67]

- The structured approach builds a paper model (data flow diagrams, process specifications) to serve as a formal set of documentation
- The prototyping approach assures the model will be a "working model". A "working model" is a collection of procedures that simulate some or all of the functions the user wants.

Alcott has purchased several software tools to help in the prototype approach. For example, Alcott has Executive Information System (EIS) tools such as Commander (EIS) and EASEL, that could facilitate the User Interface process (screen generation). Database management facilities have already been suggested. It is significant to note however, that the prototyping approach ends at the design activity of the structured

---

66 Ibid., p. 60.

67 Ibid., p. 62.

approach, because the prototyping approach is not intended to be an operational system, but intended soley as a means of modeling user requirements.

Suggested Developmental Approach

It is proposed that Alcott combine the above 2 approaches in the following manner:

- Prototype Approach
  - -- produce a fully functional prototype of the Critical Factors Module with the basic user interface, database and model manager capabilities.
  - -- provide minimal "bells and whistles" such as extensive reporting and graphics options, but enough to support the executive team
- Structured approach
  - -- incrementally develop the other modules using the above defined activities. The feedback received from the prototype should clarify the executive team's environment and real-world activities.

It should also be noted that the learning curve for using prototyping software tools will be quite high, since Alcott has no real "experience" in EIS related tools.

4.2 Future Developments

Some future PMDSS implementations to consider are:

- provide a remote communications link to allow the executive team to access PMDSS from their homes, since time has been defined as a critical success factor.

- provide other external /internal database interfaces
- add and improve modules (for example, a Distribution PlanningModule could be added to determine the efficiency of production and product distribution schedules)[68]
- merge the decision environment with the the expert system capability has been defined on the user's "wish list", but the exact details are beyond the scope of this study. However, one proposed architecture (depicted in 4.2-1) would enable PMDSS to handle both the optimization aspects (maximize pfofits) as well as incorporate different kinds of knowledge (managerial intuition, analyst expertise in solving mathematical optimization, etc...).[69]

---

[68] Thomas W. Knowles, Management Science, Building And Using Models (Irwin, Homewood, Illinois, 1989), pp. 721-722.

[69] Singh, Cook, and Corstjens. "Note: A Comment on Hybird Knowledge-Based System for Allocating Retail Space...," Interfaces , Volume 19, Number 6. (November-December 1989), p. 105.

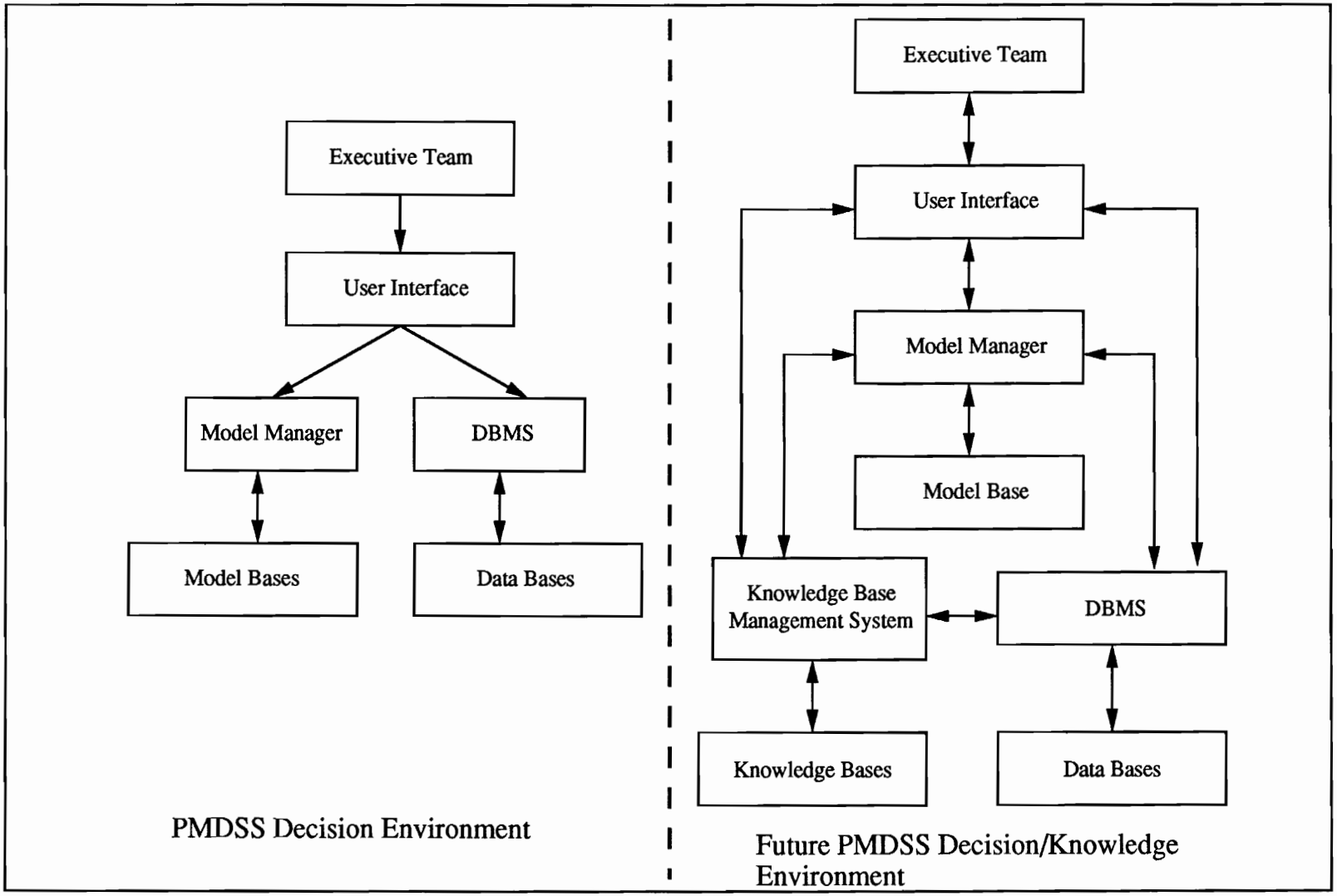


Figure 4.2-1 Decision Environments

Source: Singh, Cook, and Corstjens. "Note: A Comment on Hybird Knowledge-Based System for Allocating Retail Space...," Interfaces, Volume 19, Number 6. (November -December 1989), pp. 105-106.

References and Bibliography

1. Anderson, Michael Q., and Lievano, R. J. Quantitative Management; An Introduction. Second edition. Kent Publishing Co., Boston, Massachusetts, 1986.

2. Austin, Larry M., and Burns, James R. Management Science: An Aid for Managerial Decision Making. Macmillan Publishing Company, New York, Collier Macmillan Publishers, London, 1985.

3. Blanchard, Benjamin S., and Fabrycky, Wolter J. Systems Engineering And Analysis. Prentice-Hall , Inc., Englewood Cliffs, New Jersey, 1981.

4. Burns, James R., and Austin, Larry M. Management Science Models and the Microcomputer. Macmillan Publishing Company, New York, 1985.

5. Date, C.J. A Guide to DB2. Addison-Wesley Publishing Company, 1984.

6. Date, C. J., White, Colin J. A Guide To SQL/DS. Addison-Wesley Publishing Company 1989.

7. Eisner, Howard. Computer-Aided Systems Engineering. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

8. Gordon, Geoffrey. System Simulation. Second edition. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1978.

9. Hogan, Rex. A Practical Guide To Data Base Design. Prentice Hall, Englewood Cliffs, New Jersey, 1990.

10. IBM. "Operating System/2 Extended Edition Briefing," Rockville, Maryland, November 1988

11. IBM. SQL/Data System: Database Planning and Administration for VM/System Product and VM/Extended Architecture System Product. Version 2, Release 2, 1988.

12. Jensen, Randall W. and Tonies, Charles C. Software Engineering. Prentice-Hall, Inc. Englewood Cliffs, New Jersey 1979.

13. Knowles, Thomas W. Management Science, Building And Using Models. Irwin, Homewood, Illinois, 1989.

14. Kroenke, David M. and Dolan, Katheleen A. Database Processing: Fundamentals. Design And Implementation . Third edition. Science Research Associates, Inc., 1988.

15. Markland, Robert E., and Sweigart, James R. Quantitative Methods: Applications to Managerial Decision Making. John Wiley and Sons, New York, 1987.

16. Martin, James. DB2: Concepts, Design, and Programming. Prentice Hall, Englewood Cliffs, New Jersey, 1989.

17. Page - Jones, Meilir. The Practical Guide to Structured Systems Design . Second edition. Yourdon Press, Prentice Hall Building, Englewood Cliffs, New Jersey, 1988.

18. Ross, Steven C., Perlesky, Richard J., and Doney, Lloyd D. Developing and Using Decision Support Applications. West Publishing Company, St. Paul, MN, 1988.

19. Shaer, Sally and Mellor, Stephen J. Object-Oriented Systems Analysis; Modeling the World in Data. Yourdon Press, Prentice Hall Building Englewood Cliffs, New Jersey, 1988.

20. Singh, Cook, and Corstjens. "Note: A Comment on A Hybrid Knowledge-Based System for Allocating Retail Space..." Interfaces Volume 19, Number 6. (November-December 1989) 104-106

21. Sprague, Ralph H. Jr., and Carlson, Eric D. Building Effective Decision Support Systems. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

22. Sprague, Ralph H. Jr., and Watson, Hugh J. Decision Support Systems; Putting Theory Into Practice. Prentice Hall, Englewood Cliffs, New Jersey, 1986.

23. Thierauf, Robert J., and Klekamp, Robert C. Decision Making Through Operations Research. Second edition. John Wiley and Sons, Inc., New York, 1975.

24. Trueman, Richard E. Quantitative Methods for Decision Making in Business. Dryden Press, 1981.

25. Turban, Efraim, Decision Support and Expert Systems: Management Support Systems. Second edition. Macmillan Publishing Co., New York, Collier Macmillan Publishers, London, 1990.

26. Whitaker, David, OR on the Micro, John Wiley and Sons, Chichester, 1984.

27. White, William H., IBM, Dallas, Texas. Telephone Interview, February 1990.

28. Weaver, Audrey M. Using the Structured Techniques; A Case Study, Yourdon Press, A Prentice-Hall Company, Englewood Cliffs, New Jersey 1987.

29. Yourdon, Edward. Managing The System Life Cycle. Second edition. A Prentice Hall Co, Englewood Cliffs, New Jersey, 1988.