Concurrent Optimization in Designing

For Logistics Support

by

Melanie L. Hatch

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Management Science

APPROVED:

Ralph D. Badinelli, Chairman

Philip Y. Huang

Lance A. Matheson

Laurence J. Moore

Subhash C. Sarin

May 1994

Blacksburg, Virginia

Concurrent Optimization In Designing For Logistics Support

by
Melanie Hatch

(Abstract)

The military community has considerable experience in the areas of procuring and managing large systems. These systems are often expected to perform their intended function over a period of several years and as a result, they will require an extensive support structure consisting of personnel, equipment and spare assets. For this reason, Logistics Management has always been an important field within the military and is gaining recognition within private industry as well.

The evolutionary process which starts with the identification of a need and continues through design, production and retirement is known as a product's life cycle. Studies have shown that the decisions which are made initially, during the design of the product, will determine 80% of the total system costs. Several efforts have been initiated to improve the product design process and emphasize the life cycle approach. These include; Concurrent Engineering, Logistics Support Analysis (LSA) and Quality Function Deployment (QFD).

These efforts necessitate an overhaul of the decision-making methods used in the the product design process. Consequently, within the military community and private industry, the time-honored sequential-hierarchical-decision approach to design is being replaced with concurrent decision-making. The sequential process of the hierarchical method can lead to suboptimal designs which significantly increase manufacturing and follow-on support costs.

Abstract

Studies have suggested that long-range research efforts need to be directed to the area of integrated logistics. In order to satisfy the overall logistics objectives, there must be a full appreciation of the impact that design decision have on operations, maintenance, transportation and supply. This dissertation specifically addresses the links between product configuration, component choice, design of the manugistics (manufacturing and logistics) system and operational control policies in terms of production costs, support costs and system availability.

The goal of Concurrent Engineering is to consider the links mentioned above and incorporate them into the decision-making process during product design. Optimal product design, process selection and operating policy solutions can not be discovered unless the links among these decisions are quantified at the design phase of the life cycle. Concurrent optimization achieves this by addressing all of the decision variables associated with each stage of a product's life cycle and determining their optimal, or near-optimal, values at the design stage. This research presents a model which carries out such an optimization.

The contributions of this modeling approach are outlined as follows;

1) A model is constructed which links together the product design stage, the manugistics system design stage and the operating policy of a product's life cycle.

2) An optimization scheme is applied to the overall model structure to concurrently optimize the objective function criteria.

3) The final solution computed by the model is based on a bi-criteria value function formed from the two summary performance measure, Life Cycle Cost and System Availability.

Abstract

Dedication

I would like to dedicate this research to my parents, Bernard and Mona Hatch, whose encouragement and support made this effort possible.

Acknowledgements

My sincere thanks to my chairman, Dr. Ralph D. Badinelli, for the guidance and support he has provided throughout my stay at Virginia Tech. His encouragement and assistance has made the doctoral program a very rewarding experience.

I would also like to thank Dr. Lance A. Matheson for his insightful comments and advice throughout my dissertation. His support and inspiration has been greatly appreciated.

Furthermore, I would like to thank Dr. Laurence J. Moore, Dr. Philip Y. Huang and Dr. Subhash C. Sarin for the time and effort they spent as members of my dissertation committee.

In addition, I would like to thank my sisters and brother, Michelle, Mark and Melissa for their support throughout my scholastic endeavors.

Finally, I would like to thank Dr. Bernard W. Taylor, III, Head of the Department of Management Science, for the financial support, and the research facilities he has provided throughout the doctoral program.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The military community has considerable experience in the areas of procuring and managing large and complex systems.[1] These systems are often expected to perform their intended function over a period of several years and as a result, they will require an extensive support structure consisting of personnel, equipment and spare assets. For this reason, Logistics Management has always been an important field within the military and it has been gaining recognition within private industry as well.

The evolutionary process which starts with the identification of a need and continues through design, production and retirement is known as a product's life cycle. According to Chapman [1992], the decisions that are made initially, during the design of the product, will determine 80% of the total system costs. Any optimization with regard to cost, after the product design is completed, will affect only 20% of the total expenditure. Because of this, careful consideration must be given to addressing the impact that design decisions will have on the rest of a product's life cycle.

[1]A system is defined as a product or process that performs a specific function to satisfy the needs of the consumer. The terms 'system' and 'product' will be used interchangeably.

Several efforts have been initiated to improve the product design process. These include; Concurrent Engineering, Simultaneous Engineering, Integrated Logistics Support (ILS), Logistics Support Analysis (LSA) and Quality Function Deployment (QFD). These will be defined in detail later in the paper. All of these ideas emphasize a life cycle approach to systems design by encouraging communication between the people involved with each phase of the development.

Product Life Cycle

The life cycle of every product can be separated into the stages of product design, process design, production, operation and finally, retirement. While most flow diagrams represent these stages as distinct phases, in reality they will interact and quite often, overlap each other. Decisions that are made during one stage of the life cycle will frequently have a significant impact on the other stages. It then becomes important to address manufacturing, support or other issues, normally associated with later life cycle stages, as early in design process as possible.

A product can range from a simple consumable item, such as a nail, to a disposable item, such as a light bulb, to a repairable item, such as an automobile. It is not necessary for a product to be hardware related, it could also be service oriented like a management information system. For the purposes of this research, the products are assumed to be repairable and therefore require some type of support structure. Repairable purchases in the military would include everything from weapon systems to computers and in private industry, they might include manufacturing equipment to copiers.

The design portion of the life cycle begins after a need for the product has been identified. There are two different ways that this can occur, either the manufacturer proactively forecasts the need based on an analysis of the market, or he responds to the request from the customer.[2] [Patton, 1986] The results of the needs analysis will determine the course that the development, manufacturing and later life cycle stages will take.

There are many reasons for initiating the steps of new product development. These steps are shown in Figure 1.1. During the operational phase of its life cycle, an existing product should be assessed by the user to determine its capability of meeting the current mission requirements.[3] Data for some type of performance criteria, such as availability or support costs, are collected and compared to the current goals or specifications of the mission. If the user decides that there is some deficiency in performance, he may elect to modify the product or discard it in favor of a new design.

Quite often, the decision of whether to discard or modify the current product is based on economic factors. The user must determine if it is more cost effective to retire the existing product and purchase a new one, or if a modification will significantly enhance product performance. The decision to discard or modify the current product might occur because the mission requirements have changed, the product has been rendered obsolete by some new technology or safety considerations are involved.

---

[2]The term 'he' will be used in the generic sense throughout this paper.

[3]The term 'mission' is defined as the required function that the product must perform to satisfy the needs of the customer.

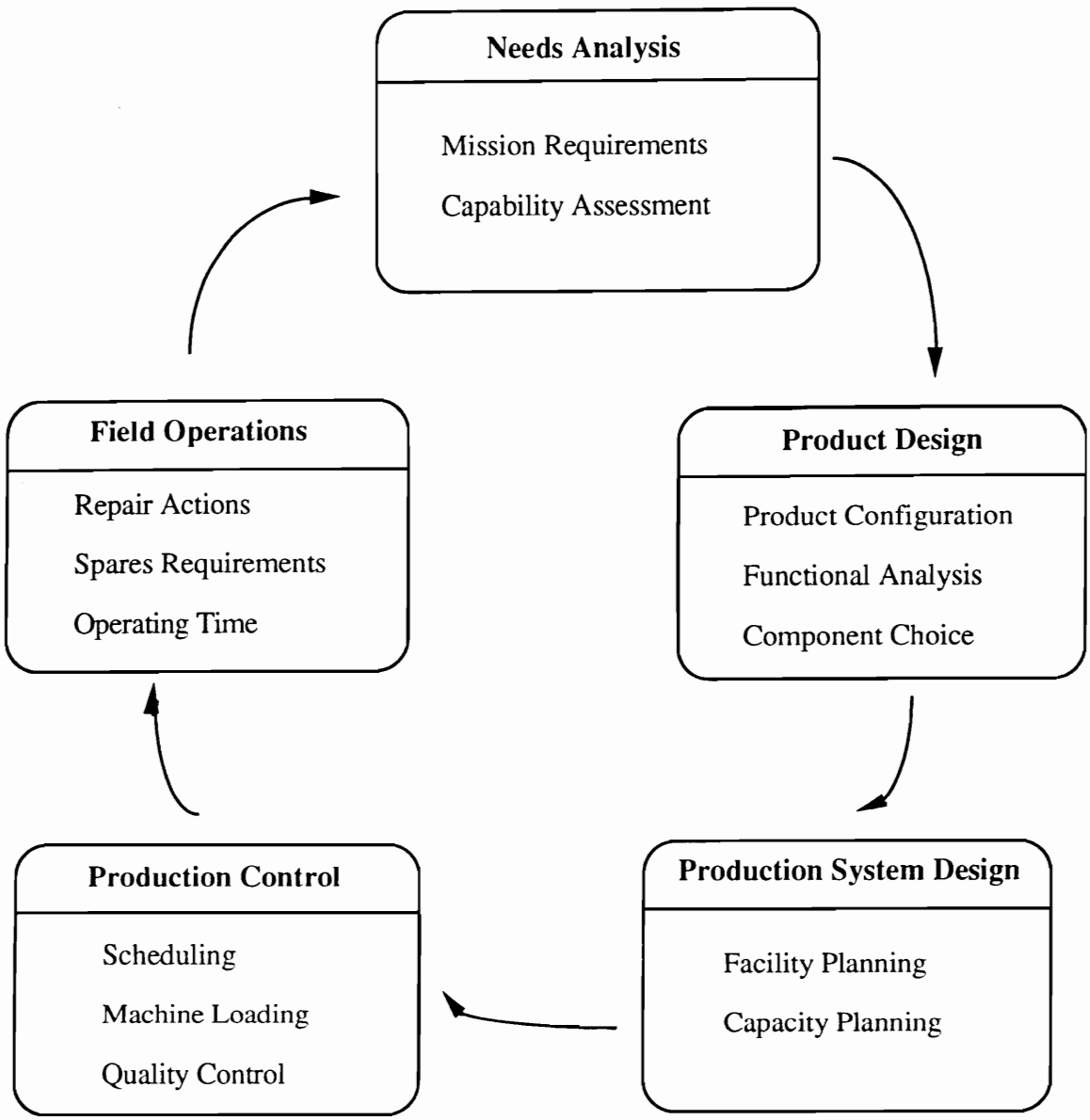


Figure 1.1. Product Life Cycle

Once the need for a product has been identified, a feasibility analysis should be conducted to determine if the technology exists to meet the specifications. Some of the steps of the analysis are; identify all of the potential alternatives that will meet the requirement, evaluate the candidate designs in terms of effectiveness measures (performance or cost) and then prioritize the list. Any alternatives which obviously violate the performance or cost constraints should be eliminated. [Blanchard,1992]

The purpose of the feasibility analysis is to define the product operational requirements, develop the product maintenance concept and identify the product configuration that stays within the constraints of technology and resources. If the current technology can not meet the needs, then new research should be initiated. [Blanchard,1992]

Operational requirements, such as minimum system uptime, must be specified by the user or forecasted by the manufacturer before the design can be accomplished. Potentially, this will dictate the configuration of the product and the specifications within which it must operate. The anticipated length of the system life should be identified as well as how and under what conditions the product will be used.

The primary and alternate missions of the product must be determined by the user. Physical parameters such as weight, volume, accuracy, speed also need to be addressed. The user must relate to the designer which physical constraints are absolute requirements and which are merely targets. For example, weight and volume might be absolute constraints for a unit that must fit within the confines of another fielded system.

Other outcomes of a feasibility study are the identification of support equipment, facilities and personnel requirements. These preliminary results can inform the user of the possibility that the current support structure will be adequate for the new design or provide ample leadtime to modify the existing support structure if necessary. The date when the new product is expected to be fully operationally should also be addressed at this point because the design approach for the new product may depend on the time available before final delivery. [Blanchard, 1992]

All of these results from the feasibility analysis can be directly translated into design specifications to be delivered to the manufacturer. The specifications can relate to the product as a whole or they may be directed at individual sub-modules of the design. They can be absolute requirements or they can be goals that the designer should try to achieve. [Patton,1986] Once the requirements have been formalized, the product design can progress to a more detailed level.

One of the initial steps during product design is to perform a functional analysis of the system. The user should determine exactly what is to be expected of the product in functional terms. This relates to what the product is supposed to do as opposed to how the product will do it. Defining the functional requirements is important as it can have an impact on product configuration as well as the operational support system that will be implemented later.

The results of the functional analysis will translate into a system functional flow diagram. This illustrates how sub-systems of the product must functionally relate to each other. This will help keep extraneous sub-systems out of the product design and serves as a guideline for a systematic

approach to the design. [Blanchard,1990] An example of a flow diagram is shown in Figure 1.2.

System functional analysis is an iterative process. As more specific information becomes available about how the system should perform, the functional diagram and associated design specifications will be updated. Eventually, this will lead to identifying the various components and piece parts that will make up the product. Once this has occurred, allocation of performance measures such as, reliability and maintainability down to the sub-system and component level is carried out.

Performance measures which relate to the system as a whole are called 'top level' measures. They must be determined by the customer in accordance with their specific mission requirements. For example, a highly critical sub-system on the product may require a 99% availability and a maintenance turn rate of one day while a less critical sub-system may need only an 85% availability to successfully perform its mission.

These system level performance measures need to be translated into lower level requirements which will dictate the product design. Several techniques are available for allocating these measures to the sub-system and component level. Preliminary logistics support factors should also be addressed at this point regarding the amount of equipment and personnel necessary to achieve the desired level of support performance.

When designing a new product, the manufacturer needs to be concerned with more than the end result, he should also consider his capability to produce the item. The design process can be

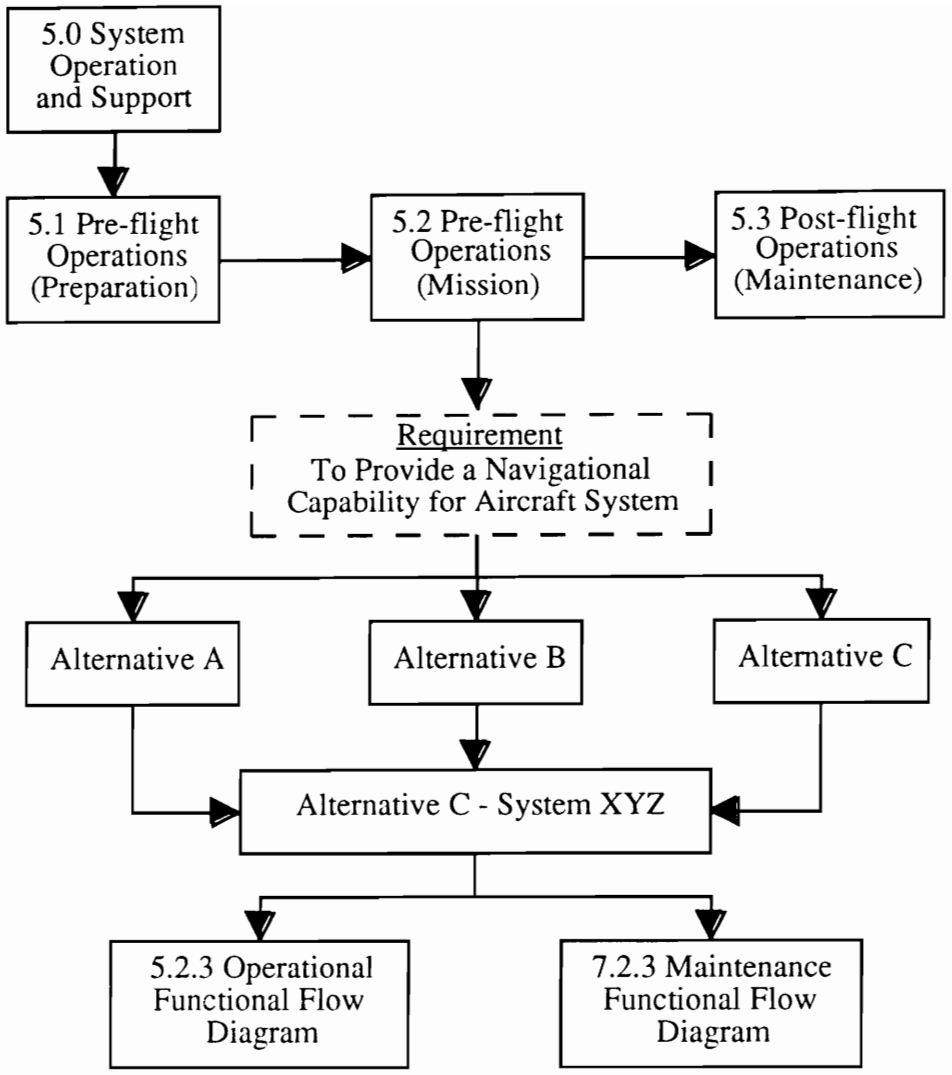


Figure 1.2 Functional Flow Diagram
Source: Logistics Engineering and Management, Blanchard, 1990

separated into two different activities, that of product design and that of process design. The most efficient and streamlined design will not be of any use unless it can be manufactured. Consideration must be given to the production system requirements prior to the final stages of product design.

Production system design includes process selection, job design, tooling design, capacity planning, facilities layout, and if necessary, facilities location planning. Constructing a production system that minimizes the time interval from design to delivery will help promote good customer relations and can also prevent another manufacturer from introducing a similar product into the market first.

There are many things to consider when designing the production system. One such consideration is whether to produce certain components or sub-assemblies at a manufacturer's own site or to sub-contract out portions of the work. Trade-off studies can be performed to determine which approach will result in the greatest benefit. Delivery schedules for outside suppliers, of both raw material and finished goods, must then be coordinated to enable a smooth assembly of the final product without delays or interruptions.

The manufacturer must also consider if his current operations have the capacity to support the production or if additional facilities, personnel or other resources must be procured. The final delivery date, as determined by the customer, will impact the decision to purchase additional equipment or hire new personnel. Certain portions of the operations may have to be modified to be able to produce the new product.

A prototype of the product may be developed to test both the product and the production system design. This allows the manufacturer and the customer to validate the actual capabilities of the product before a commitment is made to releasing the product to full scale production. This will help prevent costly rework after delivery of the product.

Once planning for the production stage has begun, the next stage in the product life cycle involves operations control. This can be divided into two separate portions as well. The first part includes the operations support of the production facilities while the second part involves the logistics support of the product during distribution and deployment.

Production control includes managing the flow of raw materials, WIP and finished goods. For the components or sub-assemblies that are produced internally, a production schedule must be determined. The assembly line workload should be balanced to smooth out the production operations. Individual jobs need to be scheduled on the various machines and the job order determined to maintain an efficient utilization rate of all the equipment. The job order will impact the layout of the equipment within the production facility. Similar jobs may be grouped together along with the required materials for assembly.

The type of production and inventory control system, such as, kanban or Materials Requirements Planning (MRP) must be determined. This can affect the availability of components that may be produced later as spares to support repair of the end product. This choice of control system is in turn influenced by the production facility.

The type of maintenance policy for the support equipment and production machinery should also be considered. If there is sufficient down time to allow for scheduled maintenance so the product assembly time is not affected, this will lead to one type of policy. If the equipment provides a warranty period, then another type of policy may be required. The skill level of the internal maintenance staff will also have an effect on the maintenance policy.

Some decisions involving field operations control can be initiated prior to the delivery of the final product. Initial provisioning is a process that is used by the military and its supporting contractors to ensure that necessary spare parts are in place when the final product is delivered. The list of items to be used as spare parts is determined by the equipment specialists. Together with the design engineers, they predict the expected number of failures for a given operating period, which then determines the initial spare parts requirements. This is essential for long lead time items - so they are available from the start of system operation.

Once the initial sparing decisions have been made, the follow-on sparing process must be determined. The location and quantity of spares to be procured should be decided by the user before actual delivery of the product. Spare parts ordering policies must be determined to ensure uninterrupted use of the product in the field.

Product repair decisions often accompany the spare parts decisions. The location of maintenance facilities will often determine the location and quantity of spare parts. Repair facilities are often classified by the type of maintenance they can perform or the skill level needed to complete the repair. The level at which a broken part will be repaired, can determine the quantity of spare

parts to keep on the shelf at a given location. The total operational time during the repair cycle is another contributing factor in determining the quantity of spares.

The operations and maintenance phase of the life cycle is the longest and can be the most costly. If logistics issues are not addressed during the previous stages of the cycle, it can result in extremely high support costs for the customer. Excessive maintenance or lack of spare parts will cause down time for the system which translates into loss of mission capability or product utilization.

System phase-out or retirement is the final stage of the product life cycle, unless the system is replaced prior to full completion of the expected lifetime. The method, and associated cost, for disposal or perhaps sale of the product should be considered as part of the total life cycle cost. The overall projected useful life of a product can be an important factor in evaluating alternate design solutions.

Logistics Measures

Although different costs and logistics measures may be associated with each stage of the product life cycle, the fact that the stages are linked can not be ignored. Many of the factors that influence the later stages should be considered during the initial design efforts. Often decisions made at the beginning, will determine what support can be offered later in the operational stages.

For example, the type of warranty for certain components, or for the entire product, can effect the computed amount of initial sparing or determine at which level maintenance will be performed during the warranty period. Decisions regarding the product configuration will also effect the level of maintenance required to perform repairs. These decisions will be made prior to the final product delivery date, and yet, they impact field support processes well beyond.

Component standardization and choice of manufacturer is another issue that will affect subsequent support of the product. Standardization can affect the level of effort required to perform certain repair actions and it can also affect availability by allowing components to be substituted for each other during repair. The manner in which the manufacturer produces the components can dictate the type of inventory policy used to maintain on-hand stock.

The customer may define several effectiveness measures that will serve as standards for acceptance of the final product. These measures can be quantified by the amount of money it will take to build them into the product. Some of these measures may be flexible, while others are necessary for safety or mission requirements. The customer may modify the acceptance criteria for these measures as more information becomes available during the design effort.

Reliability

It seems that almost every manufacturer mentions the word 'reliable' somewhere in advertising a product. This measure has become very important to consumers, given the current economic conditions. The most common definition of reliable is that the product is available when it is

needed. Technically, the definition is extended to include the requirement that the product should perform for the specified length of the mission, given that it is being operated under the conditions for which it was built.

Generally, the customer may know what top level reliability is required for the entire system. The designer must then take the overall system-level measure, and translate it into individual sub-system and component-level reliabilities. This involves starting with the overall system requirement and allocating lower level reliability goals to the sub-systems and components. These goals are computed so that when the individual component reliabilities are combined into the system measure, they will meet the top level requirement.

Quite often the designed system reliability does not equal the actual reliability experienced during field operations. There may be external factors which contribute to system failures that may not be the fault of the system. If is being operated outside of specified environmental conditions or if human factors are involved, then the resulting system failures are not the fault of the design. For this reason, failures are often categorized into inherent, induced and no defect.

Inherent failures are system breakdowns that can be attributed to either the design or components of the system. This type of failure mode is used to compute the 'designed reliability' of the system. Induced failures can be traced to some kind of problem external to the system. Perhaps the system was dropped or damaged in some other way by a force outside of the system. No-defect failures are classified as those that do not result in a maintenance action. Either a module or component which was sent to the repair site resulted in 'testing ok' or it was removed to

facilitate the repair of another part. 'Operational reliability' generally refers to the actual reliability experienced during field operations and is computed using the all of the above failure types.

Maintainability

Maintainability is another logistics measure which may be specified by the customer during the initial design stages. This often refers to the level of effort required to restore a broken asset back to serviceable condition. Even the most reliable systems will require unscheduled maintenance and then ease of repair becomes an important issue. Several logistics measures are associated with the overall concept of maintainability.

Repair Cycle Time (RCT) measures the amount of time necessary to locate and remove a failed part, repair it and return it to serviceable stock. It is necessary to accurately measure or estimate this time, because system operating time during a repair cycle determines the number of spare assets that should be kept on-hand to maintain the overall system in a operationally ready state.

While the repair cycle shows how long it takes to repair all of the failed assets, the turn rate refers to the percent of broken parts that can be repaired during a given amount of time. The turn rate may be expressed, for example, as the desire to have 50% of the failed assets restored to serviceable condition within two days. The turn rate will be influenced by the availability of support equipment, maintenance facilities and personnel to complete the repair action.

Level of repair analysis should be performed at some point during the design. This type of analysis determines the most cost effective maintenance policy for the proposed system in terms of facilities, equipment and personnel. Level of repair refers to where (or if) the system should be repaired. Certain components or modules may be repaired at the customer's maintenance sites while others should be returned to the manufacturer. Depending on the cost of the part, sometimes the recommended policy is to discard the component and replace it with a new one.

Preventive maintenance policies for the product can also be determined at this early design stage of the product life cycle. Certain sub-assemblies may require calibration or cleaning on a periodic basis. The frequency of these scheduled maintenance actions will effect personnel, spare parts and equipment requirements.

Availability

Availability is the measure that combines the output from both reliability and maintainability. In general terms, it measures the percent of time that the system was in an operationally ready state when needed by the user to perform its mission. The overall possessed time of the system can be broken out into the amount of time that it is operational (which relates to reliability) and the amount of time that the system is not-operational (which relates to maintainability). Then availability is computed by forming a percent of uptime versus total possessed time.

There are different grades of availability measures that correspond to the reliability calculations. Operational, or achieved, availability uses the total failures in its computation. Inherent

availability is calculated using only the inherent failure type. The customer needs to be aware of the fact that operational availability may not equal designed availability.

Manufacturability

Designing for manufacturability involves creating a product design that can be produced in a cost-effective manner. Manufacturability measures how easily a system can be produced using conventional or flexible manufacturing processes without impacting the quality or performance of the product. The objective is to minimize the amount of special production tooling or materials used within the process.

The design and manufacturing engineers need to communicate about the production tasks and capacities that will be required to build the product. This communication process will help to prevent the situation where half of the final products must be scrapped or reworked because they are out of specification. The manufacturing engineers can determine if the existing production system could be reconfigured to allow for the product design to be built or the design engineers could modify the product structure to fit the product into the current production process.

The quantity and variety of parts within the design can affect the production process. Common and standard components should be used as much as possible to reduce the number of potential changes to the existing production system. The product configuration and level of connectivity of the components can impact the assembly process and so the simplest design configuration as possible should be used.

Supportability

Another measure that may be specified by the customer is supportability. This relates to how well the system can be sustained in an operationally ready status over a given period of time. It deals with the spare parts forecasting and the capability to have the correct part available when required for scheduled or unscheduled maintenance.

The operations control functions of initial and follow-on provisioning are very important to achieving a high level of supportability. Initial provisioning involves forecasting the spare parts requirements for the system prior to its being fielded. This helps to ensure that the parts are available to support maintenance from the first day that they system is in opertaion. Once the maintenance policy and level of repair have been decided, then the location and quantity of support equipment, personnel and spare parts can be determined.

Long lead-time and mission-critical components and modules should be highlighted for special management considerations. Quite often these assets comprise the majority of the support cost while they represent a minority of the total required amount of spares. These parts may need to be managed separately from the other spares inventory to avoid additional system downtime.

All of these logistics measures are important criteria for designing a new system. Since rework costs increase during each phase of the system life cycle, support issues should be discussed as early as possible. The effects of different design configurations or types of components need to be identified in the initial stages of the cycle.

Logistics and Design

Designing for supportability can decrease the amount of rework and reduce acquisition cost. It can require a major effort to correct support problems if they are not discovered until after the system has been fielded. For this reason, the military community has mandated that support issues be addressed during the initial design stages for new purchases. In private industry, many corporations are integrating the ideas of Total Quality Management (TQM) and Quality Function Deployment (QFD) into the design process.

One way to ensure that the operational, support and internal production requirements are being met is to schedule regular design reviews during the design stage of the life cycle. The design and production staff as well as the customer should be present during the meetings to evaluate the proposed design and offer guidance in their specialty areas. These reviews will provide a formal set of documentation that can be used to record decisions made by the team. If these historical records contain a 'lessons learned' section, they can provide valuable information to future design efforts. [Blanchard,1992]

Several reviews may be scheduled, depending on the state of the system at a given time. An initial review may be conducted after the feasibility analysis has been performed to verify that all product requirements have been addressed. If the review team recommends proceeding with the design, then another review can be held when the performance measures have been allocated. Prior to the start of production, a critical design review should occur to baseline the design. A system is baselined when the design addresses all of the customer's requirements and it serves

as the starting point for future enhancements. At this point, any identified deficiencies in the design should have been corrected and the design is ready to be forwarded to manufacturing. [Blanchard,1992]

Configuration management plays an important role during the design process. This function coordinates all of the changes that are made to the baselined system to ensure that they are properly documented and controlled. They identify any potential for conflict and protect the integrity of the entire system. Any changes to a component or module are screened to determine if they will affect other aspects of the system or its supportability.

Current Efforts

There are several efforts within private industry and the military community to address the product design process and its impact on logistics performance measure, such as availability or supportability. It has been recognized that support costs, along with production and acquisition costs, must be considered when evaluating alternate design options for a product. MIL-STD-13882A/2B outlines a process called Logistics Support Analysis (LSA) that provides guidance on collecting logistics data to predict future support requirements.

To be the most effective, the LSA effort should begin during the design portion of the life cycle. LSA uses various quantitative models to address logistics needs during the system life cycle and uses logistics measures as points of comparison for different product designs. [Fabrycky,1991]. All current development projects using LSA within the military community produce the Logistics

Support Analysis Record (LSAR) during the design stage of the product. The LSAR is a centralized, automated data base which contains the logistics information collected during product design.

Another effort places Life Cycle Cost (LCC) issues at an equal level with performance and delivery schedule and is outlined in the Joint Design to Cost Guide. Prior to this program in military acquisition, support costs were not an important measure of merit when evaluating competing design alternatives. Previously, the emphasis was placed on mission capability and, if the product was unreliable, more money was invested for modifications or for the purchase of additional inventory. Current economic conditions have made infeasible this solution of 'buying our way out of the problem'. The plan to control support costs must be established early in the design phase of the product.

The Weapon System Master Planning (WSMP) process was established within the Air Force community to more effectively manage large systems throughout the course of the entire life cycle. The WSMP is created at the start of the life cycle and describes the management activities that are necessary to develop and support the product from design to retirement. This document is required for all new acquisitions and is also being retroactively written for certain fielded weapon systems.

Private industry has also recognized the importance of designing reliable and supportable products. Individual consumers are demanding higher quality and longer life in the products they purchase. Products that are durable and have the flexibility to be modified or upgraded to meet

new mission requirements have become important to customers with budgetary limitations. Several movements which empasize meeting the needs of the customer have become popular within private industry.

Total Quality Management (TQM) has emerged in response to the demand for better products. To be successful, it requires that all levels of the company be involved in the quality process. Managers and employees are working together to improve the design and development process. Bringing the product to market in the shortest leadtime possible, while not sacrificing high quality, is one way to maintain the competitive edge for a manufacturer.

One important aspect of TQM is establishing and maintaining communication lines not only between internal departments of the company, but with the external customer as well. One method of doing this is Concurrent Engineering which is also referred to as Simultaneous Engineering. It's purpose is to bring together all the people (including the customer) who are involved in each aspect of the product life cycle early in the design stage. This ensures that the design can eventually be manufactured and that it also adheres to the operational specifications that were determined by the user.

QFD has been used by the automotive industry since the 1980's and is being included in the quality programs of many major corporations today. The objective of QFD is to focus on quality issues early in the product life cycle. This is done by building charts, called Houses of Quality, which illustrate the relationships between customer requirements, engineering designs and the manufacturing process.

Decision Making Framework

All of the above efforts have one thing in common, that logistics support and quality issues need to be addressed early in the design stage to reduce overall life cycle costs. Decisions that are made during the design phase can have an enormous impact on the ability to manufacture the product and then provide support after it has been fielded. This type of analysis requires a new perspective on the decision making process because it involves the simultaneous optimization of many performance measures over several different stages of the product life cycle.

The traditional engineering design technique of 'throwing it over the wall' can no longer be considered as a viable option in the design process. This sequential decision making approach segregates all of the departments that are involved in systems design. It starts with the marketing department determining the customer's requirements and then throwing them over the wall to the design engineers who formulate the design and throw it over the wall to the manufacturing engineers who build the production system and so on. There may be communication within a single department, but there is none between departments. This can result in a product design that can not be manufactured or later supported during field operations. [Chapman et al, 1992]

Concurrent decision making offers a solution to the 'throwing it over the wall' method of design. The Concurrent Engineering Directorate of the U.S. Army Communications-Electronics Command says, 'Concurrent Engineering is the simultaneous and integrated engineering of all design, manufacturing, and support aspects of a product from concept through availability. It is a teaming concept.' It advocates integrating all of the members involved in the design and

encourages communication across the departments [Chapman et al, 1992]. This is the only way to ensure that decisions being made early in the life cycle will not have a negative impact on the ability to manufacture and support the product in the later life cycle stages.

There is a parallel between the concepts of Concurrent Engineering and Quality Function Deployment. The objective of QFD is to introduce quality issues as early in the design stage as possible and then to maintain them throughout the later portions of the life cycle. As mentioned earlier, this is accomplished by building matrix-style charts called Houses of Quality which show the relationship between quality and design.

Figure 1.3 illustrates the how these Houses of Quality can be used by the manufacturer. The first chart shows the relationship between customer demands and quality characteristics. The next chart takes the quality characteristics and determines the relationship to product characteristics. The third chart uses the product characteristics and relates them to the manufacturing system. The final chart shows how the manufacturing system relates to the quality control issues. The inner portion of the chart is then filled in with data that shows the strength of the relationship described by the chart.

In contrast to concurrent decision making, hierarchical decision making starts with a global or aggregate plan and then divides it into smaller sub-problems, with increasing level of detail, and solves them separately. Hierarchical Production Planning (HPP) advocates that disaggregation of the global plan follow organizational boundaries. Figure 1.4 shows the relationship between what decision is made at each particular organizational level.

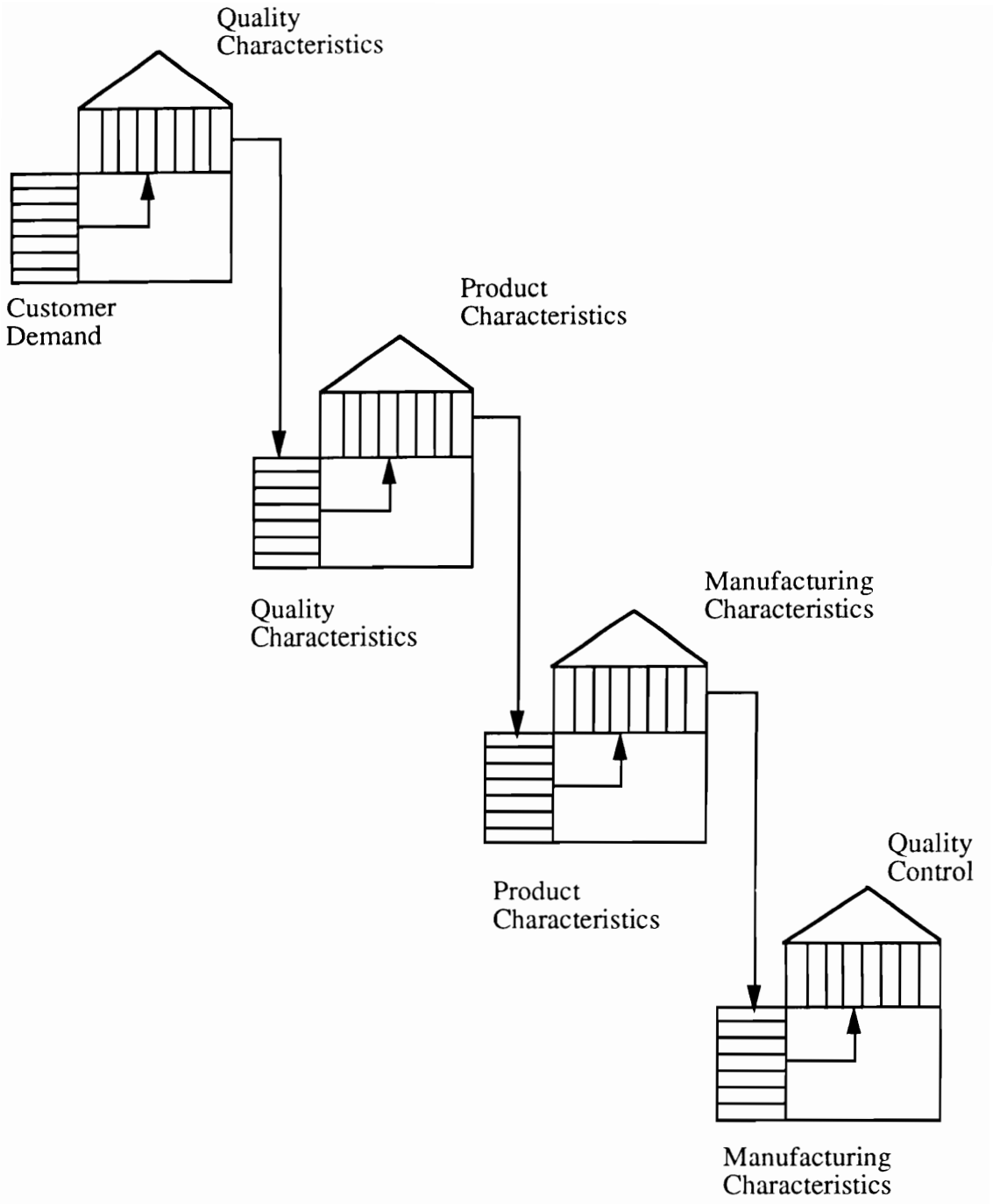


Figure 1.3. Quality Function Deployment Waterfall Chart
Source: Engineering Modeling and Design, Chapman, et al., 1992

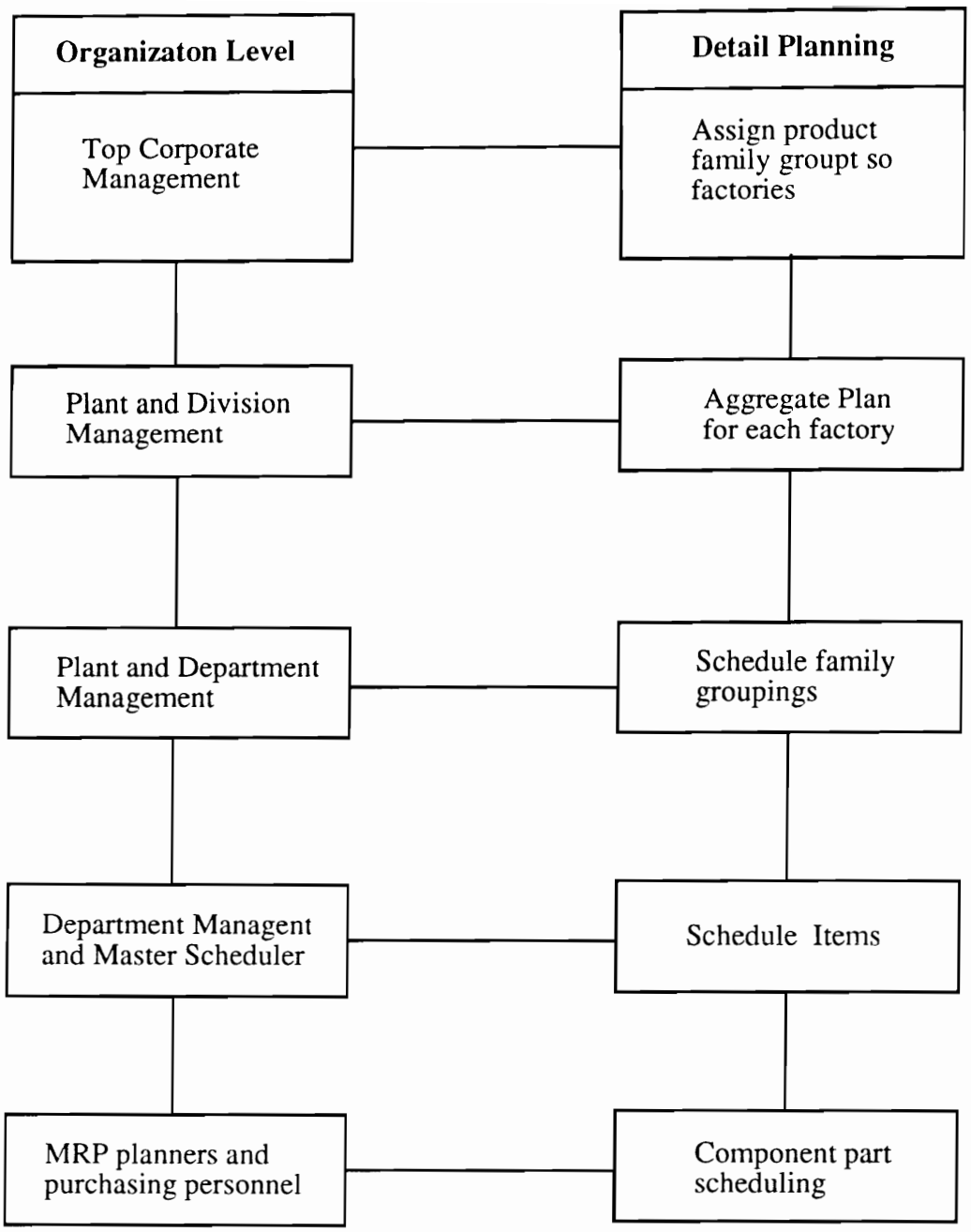


Figure 1.4 Hierarchical Production Planning

For example, if multiple production facilities are available, the first step would be to allocate the various product families to a specific facility. Once this has been completed, then the plant might determine an aggregate plan which specifies production and inventory levels for the product families it was assigned. This aggregate plan can then be used to develop a master production schedule which, in turn, will decide the component part schedule.

As you move down the hierarchy, each level of decision making increases in the amount of detail that is considered for the problem but it must also operate within the constraints imposed on it from the previous level. While optimization can be done at the various levels, quite often, there is no feedback loop to the previous level which would allow for global optimization of the entire problem. This potentially results in a sub-optimal solution for the problem.

This type of decision making is analogous to the decomposition technique of linear programming which converts large problems into smaller problems of manageable size. The decomposition procedure divides the original problem into a master problem and a subproblem. The master problem passes cost information (in a minimization problem) to the subproblem which is updated and solved using this new data. The results from the subproblem are then passed to the master problem which is revised based upon the new information.

As is evidenced by the quality and life cycle approach efforts that are happening within the military community and private industry, the hierarchical decision approach is being replaced with the concurrent decision making approach. The sequential process of the hierarchical method can lead to suboptimal results which will impact the ability to manufacture and support the

product later in the life cycle.

The trend in decision making is going toward concurrent methods and traditional hierarchical proceses are incorporating feedback loops to achieve a global optimization. Concepts, such as MRP II allow for 'what-if' type of questions to be answered or to resolve potential conflicts in the lower levels of the hierarchy. Any inconsistencies in the sub-problems are fed back to the top levels who can then revise the master plan to elminate problems.

Modeling the Decision Process

A model may be thought of as a mathematical or physical representation of an actual system which is used to provide insights into real world problems when direct manipulation of the system is too costly or complex. Models can be used to perform comparative analysis or 'what-if' analysis to evaluate potential modifications to existing systems before they are implemented or to assess the capability of an new system before it is built. Models are also used to optimize over the choices that are presented to the decision maker during the problem solving process.

Models generally contain a certain set of components which must be identified by the modeler before construction of the model can occur. Model variables are those elements which the decision maker is trying to optimize and over which he has some control. The performance measures are the criteria upon which the decision maker judges the proposed design or solution. The parameters of the model are those elements over which the decision maker has no control

but which comprise data necessary for mapping decision variables into performance measures.

For example, a service organization may wish to determine the best number of support personnel to employ or the length of time to remain open to minimize customer wait time. The variables might be; number of servers to employ, the operating hours of the organization or the number of waiting lines. Some performance measures might be; waiting time of the customers, average number of customers in line or the total cost to employ the workers. The parameters are; the arrival rate of the customers, the service rate of the employees or the maximum number of people who can fit in the facility.

Models may be classified by the type of analysis they perform or the degree to which they assist the decision maker with problem solving. Models which assume that the parameters are known with certainty are deterministic while models which allow for some uncertainty or randomness in the parameters are stochastic. The treatment of system changes over time is another means of model classification. Models which track system changes over time and adjust the variables accordingly, are dynamic while models which do not include time as a parameter are static.

The type of information that the model provides the decision maker and whether it offers feedback or advice allows for a different classification of models. Accounting or informational models may draw upon a database as an input source to determine an answer to a problem. Typically, they do not optimize with respect to the final results, they merely inform the user of the computed solution. Simulation models, as defined within the context of this dissertation, allow for some interaction with the decision by providing a 'what-if' capability or some other

means to change inputs. Decision models provide guidance or feedback to assist the user in the decision process. Usually, they operate in an interactive mode to obtain input from the user and combine this with a resident knowledge base to recommend an optimal solution.

The Weapon System Management Information System (WSMIS), which is owned by the Air Force, is one example of how these different types of models are used in the daily management of complex systems. It is composed of several sub-modules which specialize in a particular aspect of weapon system management for both peacetime and wartime operations.
Weapon system assessors must track the peacetime performance of equipment for which they are responsible to determine the preparedness of the Air Force to enter into a wartime situation. The WSMIS/Readiness Assessment Module (RAM) provides inventory and status information such as total number of available aircraft, number of aircraft down for maintenance and total number of flying hours accomplished in a given time frame. These outputs are collected by the assessor and turned into an overall readiness status for the weapon system. This sub-system is classified as an accounting model.

Air Force squadrons which deploy to another location during the wartime scenario are authorized a War Readiness Spares Kit (WRSK) which travels with them. The WSMIS/Sustainability Assessement Module computes aircraft availability and flying hour capability for these units based on the status of these WRSK kits. One of the system outputs is a list of potential problem items that are preventing the aircraft from flying their assigned hours during the conflict. The weapon system assessor, who is responsible for predicting wartime capability, must then determine if these problem items should become candidates for reliability modifications or if they should be

replaced by new equipment. WSMIS/SAM provides the capability to perform 'what-if' analysis which allows the assessor to determine what reliability is needed by the problem parts for the combat units to complete their mission. This particular sub-system is classified as a simulation model.

Another module of WSMIS is the Requirements and Execution Availability and Logistics Module (REALM). This module provides recommendations to the weapon system assessor about the optimal mix of spare assets that is needed to build the WRSK kits for wartime use. Based on the assets' demand rates and the required flying schedule of the units, this module computes the amount of each type of spare part to include in the kit to maximize overall aircraft availability for the duration of the wartime scenario. It computes two different parts lists based on an unconstrained budget or the user can input an upper spending limit and the model will recommend the best parts mix which does not violate the budget constraint.

Figure 1.5 shows a generic flow of how these different models can be used in the daily operations and management of a system. An historical set of databases are maintained which might contain maintenance data, inventory, status and utilization (ISU) of the whole system or information on the availability of individual components. This historical data will be used as input parameters for the various models. Typical accounting models will provide status, on-hand inventory levels or counts of repair actions on the equipment. MIS models will use the information to assist with spare parts ordering or identifying potential problem components and flag them as candidates for modification. DSS models could use information from the databases or from the other model types to provide guidance on scheduling repair actions to maximize system availability or

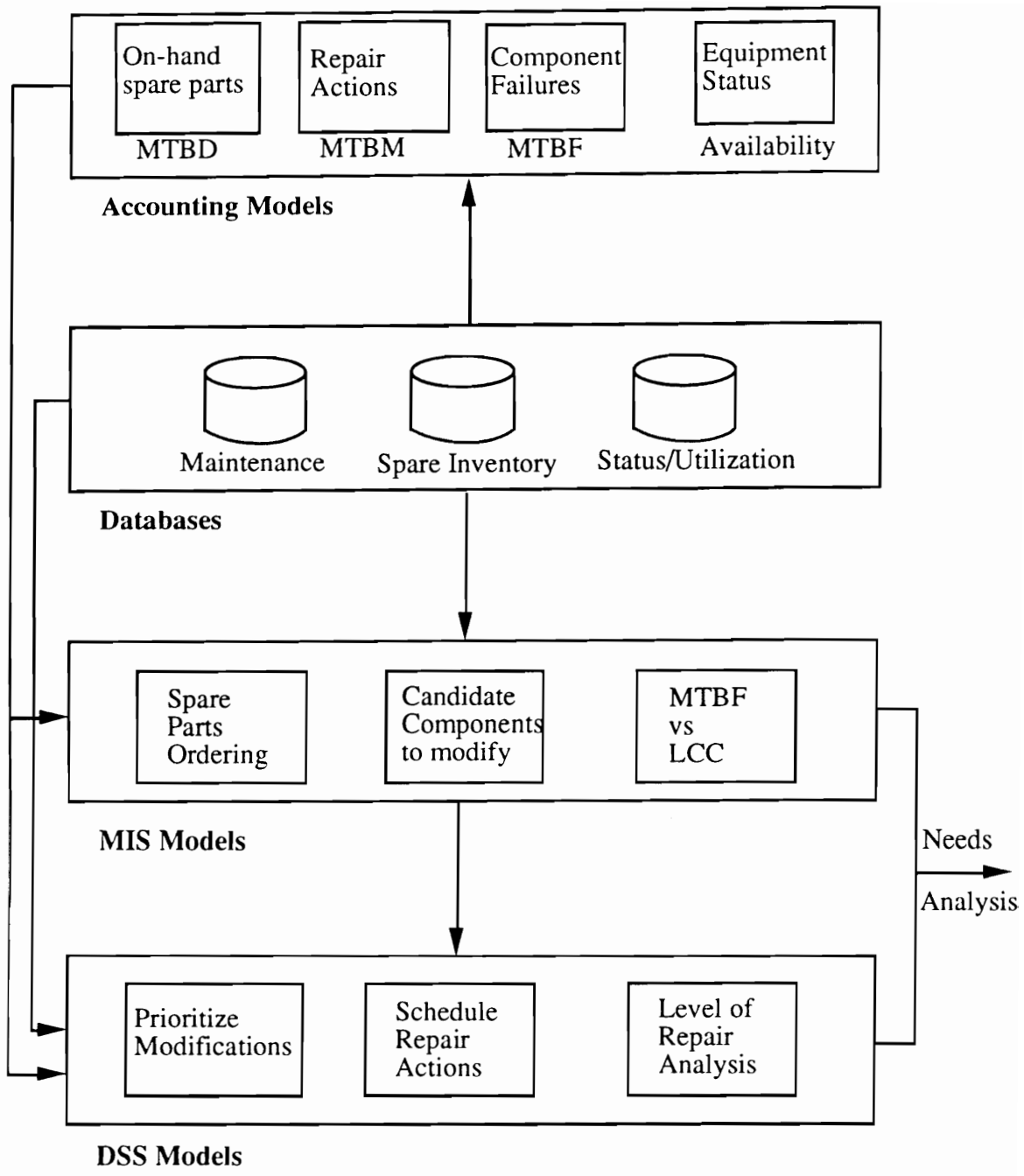


Figure 1.5. Use of Models within an Operating Location

prioritizing modifications given a limited budget. An organization could feasibly use any combination of the different model types.

Research Objectives

Many of the currently available logistics models address certain aspects of a product's life cycle, or individual performance measures but there are few that attempt to optimize the design process by recommending the optimal design strategy or by providing feedback on the impact that design decisions have on the later life cycle stages. Models which take a global view of product life cycle are necessary to accurately evaluate the performance of new products.

The following outline describes the variables, parameters and performance measures that must be considered throughout the entire life cycle of a product.

I. **Needs Analysis** - During this stage, the capability of the system to meet the requirements of the mission is assessed by the customer who then decides if he should procure a new system, modify the old one or possibly change the existing logistics support structure (buy more parts, or have more repair channels).

A. Variables
   1. Define Mission Requirements
      a. Capability Assessment
         (1) meet mission requirement - yes/no

   2. System Specs
      a. Performance

b. Logistics Requirements
    (1) MTBF
    (2) Maintenance turnaround time
    (3) Standardization
    (4) Part Commonality
    (5) Modularity
    (6) Dimensional Specs
    (7) Level of Repair

B. Performance Measures
1. System Support Costs
a. Operating Costs
    (1) Operational Availability
        (a) Induced Failure Rate
        (b) Durability
        (c) Reliability
        (d) Repairability
        (e) Maintainability
    (2) Achieved Availability
        (a) Durability
        (b) Reliability
        (c) Maintainability
        (d) Repairability
    (3) Lifetime without support
        (a) Durability
        (b) Reliability
        (c) Maintainability
        (d) Repairability
        (e) Supportability

b. Maintenance/Repair/Testing Costs
    (1) Maintainability
    (2) Reliability
    (3) Repairability

c. Inventory/Spares Costs
    (1) Reliability
    (2) Supportability
    (3) Durability

d. Distribution Costs
    (1) Transportation
    (2) Marketing/Sales

e. Facilities Costs

f. Training Costs
(1) Equipment
(2) Facilities

2. Modification Costs
a. Reliability

b. Maintainability

3. Retirement Costs
a. Durability

b. Supportability

c. Disposal

4. Acquisition Costs
a. Supportability

b. Reliability

c. Maintainability

d. Durability

e. Design specs

C. Parameters
1. Reliability

a. System Hazard Function

b. Component MTBF

c. Induced Component Failure rate

d. Operating environment characteristics

e. Uptime percentages

2. Maintainability
a. Component Failure rates

b. Repair Actions/Tasks

c. Man hrs/task

d. Repair facility utilization percent

e. Support/test equipment

f. Number of personnel

3. Supportability
   a. Component MTBD

   b. Cannibalization rate

   c. Repair facilities availability

   d. Turn around time

   e. Spares availability

   f. Storage space availability

   g. Order quantity/reorder point
      (1) Holding costs
      (2) Setup costs
      (3) Shortages

4. Durability
   a. Wear out rate

   b. Replacement rate

   c. Salvage value

II. Design Process

A. **Product Design** - Alternate designs are evaluated to determine if they will meet the needs of the customer. Based on the results of the needs analysis, the customer will specify the performance goals of the product which must be met by the design. Trade-offs between levels of performance and cost to achieve them are conducted to establish a baseline design.

1. Variables

   a. Connectivity

b. Standardization

c. Configuration

d. Component Types

e. Modularity

f. Component accessibility

g. Level of repair

h. Component repair frequency

2. Performance Measures (Conform to user specs)
   a. System Support Costs
      (1) Operating Costs
         (a) Inherent Availability
            1 Durability
            2 Reliability
            3 Maintainability
            4 Repairability
      (2) Maintenance/Repair/Testing Costs
         (a) Maintainability
         (b) Reliability
         (c) Repairability
      (3) Inventory/Spares Cost
         (a) Reliability
         (b) Supportability
         (c) Durability
      (4) Distribution Costs
         (a) Transportation
         (b) Sales
      (c) Packaging/Handling
      (5) Training Costs
         (a) Facilities
         (b) Equipment

   b. Research and Development Costs
      (1) Management/Overhead Costs
         (a) Administrative costs
      (b) Marketing costs
         (c) Configuration management costs
         (d) Data management costs
      (2) Design Costs
         (a) Reliability

(b) Maintainability
(c) Supportability
(d) Producibility
(e) Prototyping

3. Parameters
   a. Reliability
      (1) System hazard function
      (2) Component MTBF
      (3) Induced component failure rate
      (4) Operating environment characteristics
      (5) Uptime percentages
      (6) Accuracy specifications

   b. Maintainability
      (1) Component failure rate
      (2) Repair actions/task
      (3) Man hrs/task
      (4) Repair facility utilization percent
      (5) Support/test equipment availability
      (6) Number of support personnel
      (7) Component accessibility

   c. Supportability
      (1) Component MTBD
      (2) Cannibalization rate
      (3) Repair facility availability
      (4) Turn time
      (5) Spares availability
         (a) shelf life
         (b) initial spares availability
         (c) leadtime
      (6) Storage space percentages
      (7) Order quantity/reorder point
         (a) Holding costs
         (b) Setup costs
         (c) Shortage allowances

   d. Producibility
      (1) Facility layout specs
      (2) Machine utilization percentages
      (3) Inspection/testing rate
         (a) Incoming material
         (b) Finished products
      (4) Component standardization

B. **Production System Design** - The results of the product design will influence the decision made during this stage of the life cycle. The existing production system must be assessed to determine if it has the capability to produce the new product. Additional equipment, facilities or personnel may be required to incorporate the new product into the production process.

1. Variables
   a. Actual output
   b. Product standardization
   c. Lot sizing
   d. Layout type
      (1) Product
      (2) Fixed position
      (3) Combination
   e. Location of facilities
   f. Mode of transportation
   g. Make vs buy issues
   h. workforce requirements
      (1) worker skill level
      (2) training requirements

2. Performance Measures
   a. Design Costs
      (1) Producibility
      (2) Location
   b. Facilities Costs

3. Parameters
   a. Producibility
      (1) Effective capacity percentages
      (2) Efficiency percentages
      (3) Utilization rates
      (4) Safety requirements
   b. Facilities
      (1) Process policy

(a) Continuous
(b) Intermittent
(c) Project
(2) Idle/Utilization percentages
(3) Information requirements
(a) computer availability
(b) data requirements

c. Location
(1) Transportation rates
(2) Labor availability
(3) Weather requirements
(4) Locale costs

III. **Operations Control**

A. **Production Operations** - The daily operations of the production process occur at this stage in the life cycle. Given the constraints of the production system design, the decision maker must schedule the tasks to be performed, address quality control policies and determine lot sizing needed to meet the demand for the product.

1. Variables
a. Production task sequencing

b. Machine loading

c. Production lot sizing

d. Inspection sample sizing

e. Tolerable defect rate

f. Number of support personnel

g. Level of repair requirements

h. Spare parts - ordering requirements

2. Performance Measures
a. Production Costs
(1). Quality Control
(2). Producibility
(3). Facility management

(4). Equipment management
(5). Inventory control

b. Distribution Costs
(1). Transportation
(2). Packaging/Handling

3. Parameters
a. Producibility
(1) Facility layout specs
(2) Machine utilization percentages
(3) Assembly line balancing
(4) Machine setup rates

b. Quality Control
(1) Inspection rate
(a) Incoming material
(b) Finished goods
(2) Training frequency

c. Facility Management
(1) Space utilization percentages

d. Inventory Control
(1) Order quantity
(a) Holding costs
(b) Shortage costs
(c) Part usage rate
(d) Order/ship time
(2) Packaging/Handling

B. **Field Operations** - This stage closes the life cycle loop by leading back to the Needs Analysis stage. The new product is delivered to the customer who incorporates it into his existing support structure. Feedback of the actual performance data can be provided to the manufacturer for historical records or to modify any products still in the production stage if necessary.

1. Variables
a. Maintenance cycles

b. Operating hours

c. Number of personnel

d. Training schedules

e. Equipment usage

f. Maintenance task sequencing

g. Spare parts
   (1) order quantity
   (2) order frequency

2. Performance Measures
   a. System Support Costs
      (1) Operating Costs
         (a) Operational Availability
            1 Induced Failure Rate
            2 Durability
            3 Reliability
            4 Maintainability
         (b) Achieved Availability
            1 Durability
            2 Reliability
            3 Maintainability
            4 Repairability
         (c) Lifetime without support
      1 Durability
            2 Reliability
            3 Maintainability
            4 Repairability
            5 Supportability
      (2) Maintenance/Repair/Testing Costs
         (a) Maintainability
         (b) Reliability
         (c) Repairability
      (3) Inventory/Spares Costs
         (a) Reliability
         (b) Supportability
         (c) Durability

   b. Modification Costs
      (1) Reliability
      (2) Maintainability

   c. Retirement Costs
      (1) Durability

(2) Supportability
(3) Disposal

3. Parameters
   a. Reliability
      (1) System hazard function
      (2) Component MTBF
      (3) Induced Component Failure Rate
      (4) Operating environment characteristics
      (5) Uptime percentages

   b. Maintainability
      (1) Component failure rate
      (2) Repair actions/tasks
      (3) Manhour/tasks
      (4) Repair facility utilization percent

   c. Supportability
      (1) Component MTBD
      (2) Cannibalization rate
      (3) Repair facilities availability
      (4) Turn around time
      (5) Spares availability
      (6) Storage space availability
      (7) Order quantity/reorder point
         (a) Holding costs
         (b) Setup costs
         (c) Shortages

In order to successfully implement a concurrent decision making policy, a structure of the relationship between the variables, parameters and performance measures similar to the above outline must be constructed. Because of the links between the life cycle stages, the variables at one stage become the parameters of the next stage. A feedback mechanism must also be incorporated to show how decisions made at a previous stage is affecting the later stages and to provide guidance on making the decisions.

Moore [1986] has identified a hierarchy of decisions that must be addressed during the design portion of a Multiple Repairable and Logistics System (MREAL) which is shown in figure 3. He advocates taking a 'holistic' approach to modeling the MREAL systems due to the potential impact that design decisions will have over the product life cycle. This point of view has been validated by the growing trend in the military and industry toward the life cycle approach to decision making and product design. Moore built MREAL1, is his first attempt to use logistics performance measures to optimize design decisions. The focus of MREAL1 is the link between the Product Design stage and Field Operations.

Reasor [1990] extended Moore's ideas on MREAL systems by expanding the types of decisions that are included in the optimization process and by building a decision support model called CODAS (Concurrent Design Analysis System) that is more user-friendly. It considers alternate design solutions for a set of customer requirements and computes an overall weighted system performance measure based on the individual performance of life cycle cost, availability, reliability and maintainability. This overall system measure can then be used to compare the alternate designs.

The need for comprehensive models which can optimize design decisions and determine the impact that these decisions will have over the whole product life cycle has been recognized by both the military and private industry. The complexity involved with modeling the entire logistics support structure has prevented the development of these 'holistic' models. While Moore and Reasor have made an excellent contribution in this area of integrating logistics and design, there is still much to be accomplished.

The focus of this dissertation will be on quantifying the links between the Product Design stage, the Productions System Design stage and the Production Operations stage. The product design decisions will consist of product configuration choices, such as, modularity or parallel redundancy and component brand name choices having different performance levels. The production system is assumed to be of a cellular arrangement and the design decisions relate to adding new cells or equipment to existing cells. The Production Operations decisions involve assigning the production of various components to the cells and scheduling the processing of the components through the cells.

Chapter 2: Literature Review

The need for Logistics Modeling

The importance of logistics support and its impact on overall product performance has been recognized by both the military community and private industry. The military has mandated such efforts as Logistics Support Analysis and Integrated Logistics Support and most service or manufacturing companies have implemented formal Total Quality Management programs. Consumers are demanding higher quality and reliability from the products they purchase and this has forced manufacturers to take a more 'global' view of the design process.

Most of the decisions which affect the logistics support of a system occur well before it is manufactured. Studies have shown that almost 80% of the support costs for a product are determined prior to production. These decisions are the result of trade-offs between performance measures such as, reliability, maintainability and availability which then affect location of spare assets, number of repair personnel and support equipment. There is a need for comprehensive decision models which can address these issues and provide guidance to the decision maker during the design process. [Geisler and Murrie, 1981]

For complex systems, the design and production process can happen over a period of several years. As a product evolves throughout its life cycle stages, different types of information become available to the designer at various levels of detail. This introduces a need for the decision model to be flexible enough to incorporate changes that might occur because of new information. Geisler and Murrie [1981] state that a model which can be used in this fashion during the course of the entire development process would be most useful.

In his review of multi-echelon inventory systems, Clark [1972] suggests that long range research efforts be directed to the area of integrated logistics. In order to satisfy the overall logistics objectives, there must be a "full appreciation of the interplay among operations, maintenance, transportation and supply." Clark proposes a method called 'micro-simulation' to model the system which involves keeping status records that are periodically updated with new information or adding in new components to the system as it changes over time. This type of model would allow the decision maker to perform "short-term accelerated-time, simulated excursions" which would represent the results of different design decisions.

The following sections evaluate available models and categorize them within the framework discussed in chapter one. A model's merits will be determined by where it fits into the life cycle structure and how it treats the variables and input parameters. For example, a model might be categorized as an MIS model, and then further sub-categorized as a deterministic model.

Product Design

Air Force Materiel Command Regulation (AFMCR) 800-23 outlines the policy for performing level of repair analysis for new systems. The model, Network Repair Level Analysis (NRLA), uses component failure rates, number of support personnel, number and type of support equipment and repair facilities, labor rates, order and ship time of spare assets, and cost of equipment as parameters. It allows for two levels of repair, intermediate and depot and two levels of indenture, Line Replaceable Units (LRU) and Shop Replaceable Units (SRU), for the system. The decision variables are the level of repair at which the LRUs and SRUs will be repaired and the amount of spare assets to have on hand to support the number of repair actions. The performance measure used to compare the alternate repair choices is total support cost. The parameters are assumed to be known and static but it does allow for 'what-if' analysis. While this model provides maintenance policy recommendations to the decision maker, it assumes that the system design is in place and so does not optimize on the design process.

The Logistics Assessment Work Station (LAWS) was built by Dynamics Research Corporation to analyze the support requirements for new systems or modifications to existing system. It uses component failure rates, type of support equipment, repair rates, labor costs, facility and equipment costs, operating hours, personnel training costs and level of repair as parameters. It computes peacetime measures of system availability for a given operating scenario, life cycle cost assuming a twenty year life expectancy and wartime measures of capability to perform the mission without repair support and number of cargo aircraft needed to deploy the system and its support equipment. The parameters are assumed to be deterministic and dynamic. It evaluates

a specific system design and provides a user-friendly 'what-if' capability but it does not optimize the design decisions. [LAWS Functional Description, 1986]

The Logistics Composite Model (LCOM) is a Monte Carlo simulation that emulates base level operations. Aircraft are flown according to the operations schedule and sub-systems or components fail which then must be repaired or replaced by maintenance personnel. The parameters are component failure rates, maintenance repair times, number and type of spare assets and type of support equipment. The performance measure is the ability of the aircraft to meet the required flying hour schedule. This model will evaluate the capability of a type of design if all of the input information is available, although it is normally used for fielded systems.

Kaplan and Orr [1985] offer a model, called the Optimum Allocation of Test Equipment/Manpower Against Logistics (OATMEAL), which determines the optimal maintenance and spare parts stockage policies for a given system design. It evaluates the 'repair vice throw away' decision and if the component is to be repaired, then it will decide at what level of maintenance the repair should take place. The input parameters are repair equipment costs, time to repair the components, component failure rate and cost of labor for repair. It minimizes total repair and spare parts cost using an operational availability goal as a constraint.

Gross and Ince [1978] have also considered the maintenance and spare parts allocation problem for a given system design. Based on the concept of cyclic queuing, they have formulated a model which computes system availability as the performance measure. It assumes a multi-level maintenance capability and computes the number of spares required at each repair site to support

the repair actions. The models assumes an exponential failure rate for the system and an exponential service rate for the repair locations. The model then recommends the number of service channels needed to complete the repair actions along with the number of spares.

Murthy [1990], Misra and Ljubojevic [1973], Jedrzejowicz [1988] and Tapiero [1987] offer a method for determining an optimal system configuration based on meeting an overall system level reliability goal. The input parameters are the individual component failure rate distributions and the cost of improving the reliability for each component. The computed performance measure is the overall cost required to achieve the top level system reliability.

Production System Design

Shirley [1990] offers a model which selects the configuration of a cellular manufacturing process to minimize the production cost for a redesigned system. When a family of products is considered for redesign, a 'typical' product (also called the core product) which represents the whole group can be used to develop the cell assignments and task scheduling within the cell. An integer programming formulation which minimizes the time to redesign is used to select the core product from the family. Once this selection has been made, the allocation of the products to a specific cell for production is accomplished. The model calls for processing time by machine for a cell, production costs for each family in a given cell and machine capacity within a cell as the parameters. It uses total production cost as the performance measure and assumes the input parameters are know with certainty.

Chaharbaghi [1990] discusses the merits of using discrete event simulation to evaluate different production system design alternatives. Machine capacity, sequencing of tasks on the production line, reliability of equipment used in the production process and scheduling of production lots on the machines are advanced as potential input parameters. The simulation can be programmed to calculate manufacturing lead time, resource utilization, work-in-process levels or percentage of time demand is met on-time as performance measures. He concludes that simulation is one of the few analysis tools that can incorporate the complexity of today's production system operations in order to assess the capability of alternate designs.

Production Operations

The value of traditional production control performance measures such as utilization of resources, and efficiency of the system should be reconsidered, according to Kim, et al [1988] and Lea and Parker [1989]. They advocate the tracking of production lead time, quality, and inventory levels for use as the performance criteria. Wisner and Fawcett [1991] also criticize the traditional measures by stating, 'they lack the ability to fully guide the firm in its efforts to achieve manufacturing excellence.' Performance criteria should be expanded to include quality measured in terms of percent reduction in defects, flexibility measured in terms of percent increase in average number of setups, dependability measured as percent reduction in lead time per product and finally innovation measured in terms of percent increase in number of new products.

Field Operations

The Reliability and Maintainability Information System (REMIS) provides the Air Force with system status information such as operating hours, number of hours the system is fully mission capable, number of system failures, number of repair actions and length of time to perform the repair. It also computes logistics performance for component and sub-system reliability, maintainability and availability. It allows a system manager to evaluate the overall readiness of the system in order to determine if the system is meeting its mission requirements. This type of information is only useful for systems that are fielded and so it can be used during the needs analysis stage or the field operations control stage.

Berg [1984], Taylor and Rodriguez [1986], Assaf and Levikson [1982], and Ansell et al [1984] discuss models which develop optimal preventive maintenance policies for fielded systems based on minimizing overall maintenance costs. The input parameters include the component failure rates, the cost to replace the item and the cost to repair the item. The models provide guidance on when to replace components of the system to prevent unscheduled repair actions.

Graves [1988] presents a model which evaluates different repair strategies for a system that is constrained by a two day maintenance turn time. The repair site handles two different types of failure modes, one for the sub-assemblies of the system and the other for the individual components. The model provides a recommendation about the number of repair personnel and the amount of spare assets to maintain in inventory at the repair site to achieve the two day turn

time goal. Transportation time to the repair site, labor costs, repair times for the different failure modes and component/sub-assembly failure rates are needed as inputs for the model.

Dyna-METRIC, for Dynamic Multi-Echelon Technique for Recoverable Item Control, is a model built by RAND corporation that is used by the Air Force for aircraft availability studies. The model requires component/sub-assembly failure rates, mission operating times, repair cycle times, amount of on-hand inventory for spare assets and level of repair as input parameters. It allows for multiple repair sites and assumes there is adequate repair personnel to restore the broken assets to serviceable condition within the repair cycle time. It allows for trade-off studies to be conducted between cost of improving component/sub-assembly reliability as opposed to simply procuring more spare assets.

The above models were categorized based on the type of decision variables that they computed and where in the life cycle they could be used. Most of the models fit into one specific stage of the cycle and do not attempt to optimize across the stages. This type of simultaneous optimization is necessary for the concurrent decision framework to be successful for a given systems design effort. The next chapter presents an approach for establishing the links between product design, production system design and production control.

Chapter 3: Approach

The importance of optimizing the systems design process was discussed in chapter one. Since almost 80% of the total product cost is determined by decisions made during design, managers have increasingly focused their attention on this phase of the life cycle. This chapter will present a model formulation which quantifies the dependencies among the product design, production and logistics system design, production control and field operations control stages.

Product design decisions can no longer be made in a vacuum or using the sequential process that previously existed. The entire design team, and the customer, must be present from the beginning to contribute to the decision making process. The total impact of design choices must be considered as early in the life cycle as possible. This requirement forms the basis for concurrent optimization of product design, facilities design and operations control. This type of optimization is the Operations-Research methodology required by Concurrent Engineering.

Concurrent optimization addresses all of the decision variables to be considered at each stage of the life cycle and determines their optimal values in a simultaneous fashion. Optimal product design, process selection and operating policy solutions can not be discovered until the links among these decisions are quantified at the design phase of the life cycle. This research presents

a model formulation which includes these decision variables and their associated performance measures.

The contributions of this modeling approach are outlined as follows;

1) A model is constructed which links together the product design, manufacturing and logistics system design (manugistics) and manufacturing operations phases of the life cycle.

2) An optimization scheme is applied to the overall model structure to concurrently optimize the objective function criteria.

3) The final solution computed by the model is based on a multi-criteria value function formed from the individual objective functions.

The overall optimization model formulation is based on a modular design which incorporates different embedded descriptive models that are used to compute the performance measures which determine the optimal values of the decision variables. For the purpose of illustration in this research, certain descriptive models were chosen to be embedded which may not be the most accurate models for a specific type of product design. However, the characteristics that we assume of these models are quite general; consequently, other forms of these embedded models can be chosen as interchangeable modules without destroying the functional relationships of the performance measures.

First, we review the decision variables that are included in the model formulation. We divide these variables into three major categories; the product design process, the manugistics (manufacturing and logistics) system design process and the production control and field

operations control support process. The model parameters and performance measures are then discussed in the following section.

Product Design

The design decisions considered in this modeling approach involve choosing the optimal sub-assembly and component configuration, from a set of alternatives, to satisfy a given set of functional requirements for the overall product. The set of alternative configurations may include the choice of purchasing outside components and sub-assemblies or designing and manufacturing them internally.

A typical product structure will group components together to form sub-assemblies to facilitate repair actions or the manufacturing process or because they perform a specific function within the product. The preliminary design of the product will establish a set of sub-assembly and component requirements, each of which is a generic unit that is necessary for performing a specific function. An example of a sub-assembly requirement would be an amplifier with a certain gain or signal/noise ratio in an electronics assembly.

Once the component and sub-assembly requirements that are to be included in the product design have been determined, the design engineer must choose a specific brand name from a list of manufacturers which produce the type of sub-assembly or component[4]. The design engineer may

[4]The phrase 'component or sub-assembly type' is synonymous with 'type' and is defined as a specific brand name from the list of available options

also offer as an option to design and produce the component or sub-assembly type internally if the external choices do not meet performance or supportability specifications. Each external type on the list will have an associated vector of parameters, such as failure rate distribution, purchase price, manufacturing leadtime or length and cost of warranty while the internal list will be expanded to include process and setup time by type of machine or defect rate and inspection time. The full parameter list will be described later in the chapter.

The choice of component and sub-assembly types to be included in the product will affect the overall system reliability distribution and the choice of repair location for the assembly and its sub-assemblies. The system configuration will also impact the amount of spare assets to procure and the ability to produce and assemble the final product. While certain components might yield higher product costs, they could dramatically reduce the follow-on support costs during the system operations portion of the life cycle.

The product structure can be represented by a network of sub-assemblies and components connected at different levels which are subordinate to the overall system level. The total product configuration will also be called the assembly. Figure 3.1 illustrates how the various component and sub-assembly requirements might be configured for a typical system.

We define two kinds of entities in this product structure; requirements and options. A requirement is a set of functional specifications for an item. An option is a lower level configuration that can satisfy a requirement. Using Figure 3.1 as an example, requirements appear as letters in the structure and options appear as numbers. Therefore, configuration 2 is

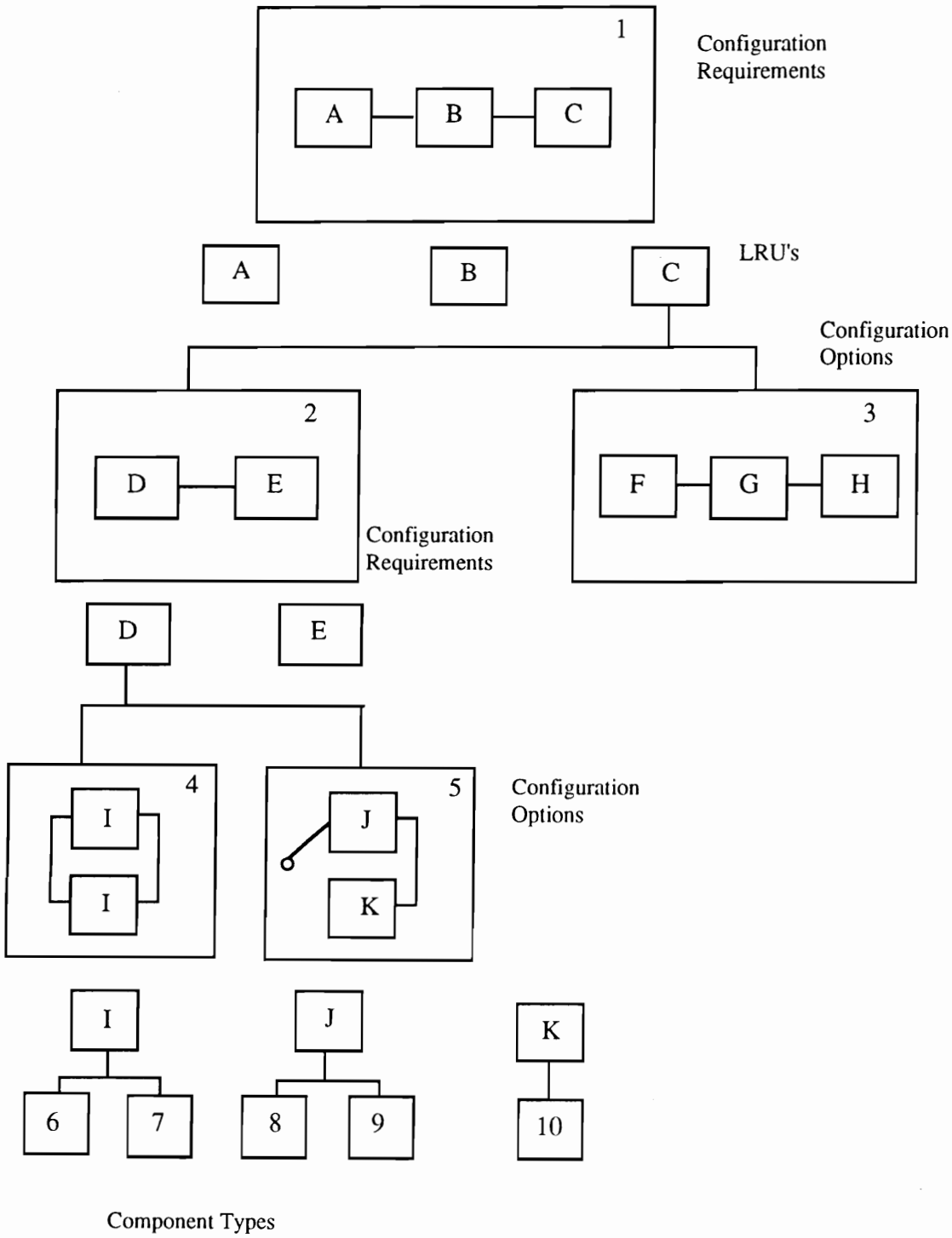


Figure 3.1. Product Structure

an option which can be used to satisfy requirement C. Individual item requirements may be grouped together to form sub-assembly configurations which, in turn, can be grouped together to form other sub-assemblies.

The overall product is configured from a set of *requirements* that is given by the design engineer. Each of these requirements has an associated set of *configuration options* which includes purchasing the unit from an outside supplier or manufacturing it internally. There may also be different ways to configure these internal designs based on their functional requirements. The set of configuration options and list of outside purchase options will be input by the design engineer. The alternating scheme of requirements at one level and then options at the subordinate level, depicted in Figure 3.1, continues until the product is broken down into individual component requirements. The final set of options then includes individual purchased components or internally manufactured components.

The product structure in Figure 3.1 consists of three different types of configurations; series, active redundant and passive redundant. The availability of a system depends on the configuration and availability of the requirements from which it is built. Option 2 shows two requirements in a series configuration. When a systems consists of requirements in series, all the requirements must be operating for the system to operate. Option 4 shows a set of two requirements in active redundancy. In this configuration, the system will operating as long as one of the individual requirements is operating. Option 5 shows two requirements in passive redundancy. One requirement is designated as the prime unit and the other is a stand-by unit. The system is operational as long as one of the requirements is operational. The difference

between the active and passive systems is that all of the requirements operate when the system operates in the active configuration, while only the prime operates in the passive configuration and the stand-by is shut down until the prime unit fails.

While a product might consist of many different configurations other than the three mentioned above, the structure in Figure 3.1 is still a valid representation of all product types. Any sub-assembly, in which all components continuously operate, can be represented by an equivalent series/parallel arrangement of its minimal path or minimal cut sets [Barlow and Proschan, 1981]. A minimal path set is the smallest set of components that are required to be operational for the system to operate. A minimal cut set is the smallest set of components that will cause the system to fail when all of the components in the set fail. The stand-by configuration is then used for products whose components do not continuously operate. Therefore, the series, parallel and stand-by configurations are the minimal number of configurations required to represent the entire set of possible product structures.

A Line Replaceable Unit is a special type of sub-assembly. It is a collection of one or more components or sub-assemblies that are grouped together to facilitate either the repair or the manufacture of the product. Sub-assemblies are designated as LRUs by the design engineer. An LRU can consist of one or more configuration requirements, and it is the smallest set of these requirements that is removed from the product upon failure in order to initiate a repair action. It will be assumed that the set of LRUs for a product are mutually exclusive in the sense that each component is contained in only one LRU. LRUs might be grouped together to form other sub-assemblies, but one LRU will never contain another LRU.

The typical product configuration will consist of 5 - 20 top-level serial requirements with each requirement consisting of 1-10 sub-assemblies and a typical sub-assembly will be made of 2-5 component requirements. It is assumed that the decision maker has narrowed down the set of possible configurations for a given requirement and will contain 2 - 3 choices for each requirement. Each component will typically have 2 or 3 potential manufacturers.

The following product design variables will determine the actual choice from the set of configuration options;

x_r = option number of the option chosen to fill requirement r from the list of available choices for requirement r

$\mathbf{Y}_r$ = set of option numbers for the options chosen to fill requirement r and all lower level sub-assemblies and components of requirement r

Throughout this dissertation, boldface type will be used to indicate set or matrix vector notation.

Manugistics System Design

Based on the values of the product design variables, the manufacturing system design decisions must consider 'make vs buy' options, assignment of components to manufacturing cells or construction of new cells if the current capacity can not meet the production requirements. The logistics design decisions consider the assignment of repair support to the sub-assemblies and the operating locations.

It is assumed that the manufacturer can also be a repair source for the operating locations and depots, so he must maintain stocks of sub-assemblies and components to fill the demand for spare assets from them. Figure 3.2 shows a layout of the manufacturing operations and the potential sources of demand for a typical product. There are cells which produce the components that are not purchased from an external supplier and there are assembly lines which build the sub-assemblies and the final product. Since the manufacturer is also a source of repair, there is a repair cell, to service incoming broken units, which can also rework defective items if necessary. A more detailed discussion of the repair and operating location structure is presented later.

The manufacturing system design variables are represented by the following list;

c_j = number of production cell which serves as the manufacturing site for component j

This variable indicates the assignment of the manufacturing cell that will produce or assemble each component or sub-assembly of the product. Processing times for each component in the cell and the capacities of the cells will be needed to make the decision of where to assign the components and sub-assemblies. An external manufacturer is also considered to be a cell for this decision variable and will be assigned cell numbers along with the internal cells.

It will be assumed that the manufacturing operations are producing other components and the list of components from this new design will merely be adding to the load on the existing cells. Any new cells that the manufacturing facility is considering for the sake of capacity expansion are added to the database of existing cells and these new cells are given an initial load of zero.

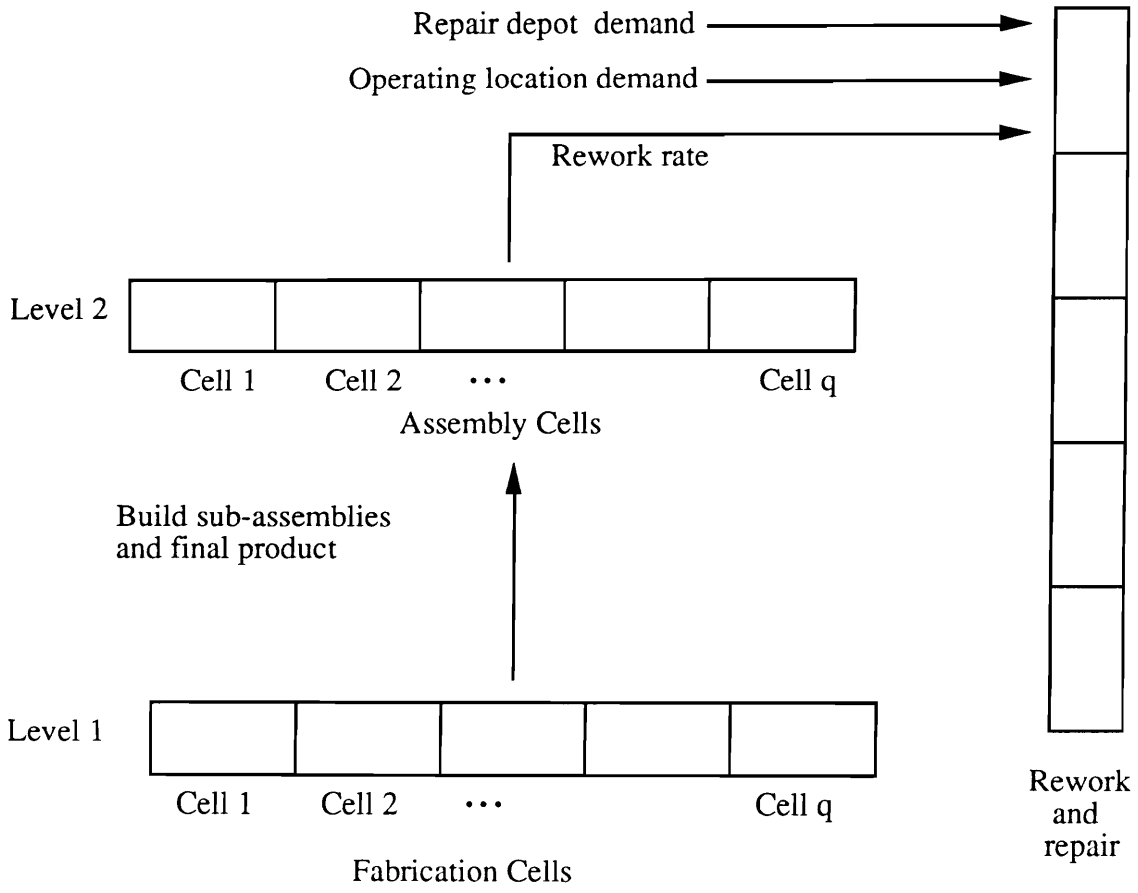


Figure 3.2. Manufacturing Operations

The other portion of the manugistics design involves the support structure for the operating locations that are using the product which is illustrated in Figure 3.3. Each location may perform certain maintenance actions at their own repair facilities or they may send sub-assemblies and components back to a depot or warehouse to accomplish the higher level maintenance tasks. For repair of warranty items, or complicated failure modes, the system may be returned to the manufacturer for repair. The logistics decision variable is defined as follows;

$m_{\ell n}$ = The location of repair at which LRU ℓ, from operating location n, should be repaired.

This variable determines whether or not a configuration option should be repaired upon failure and if it should be repaired, the level and location at which repair will be carried out. It will take on the value of zero or the repair location number. A value of zero for this variable indicates that configuration option ℓ should be discarded upon failure.

It is assumed that a particular repair facility is configured to repair only certain families of sub-asseblies. This happens when unique equipment or labor skills are necessary to perform the repair actions. This can reduce the number of alternate repair sites in which to repair a sub-assembly and its lower level sub-assemblies.

Operations Control

Operations control decisions refer to both production-control and field-control decisions. Production-control decisions involve determining the production quantities for the components

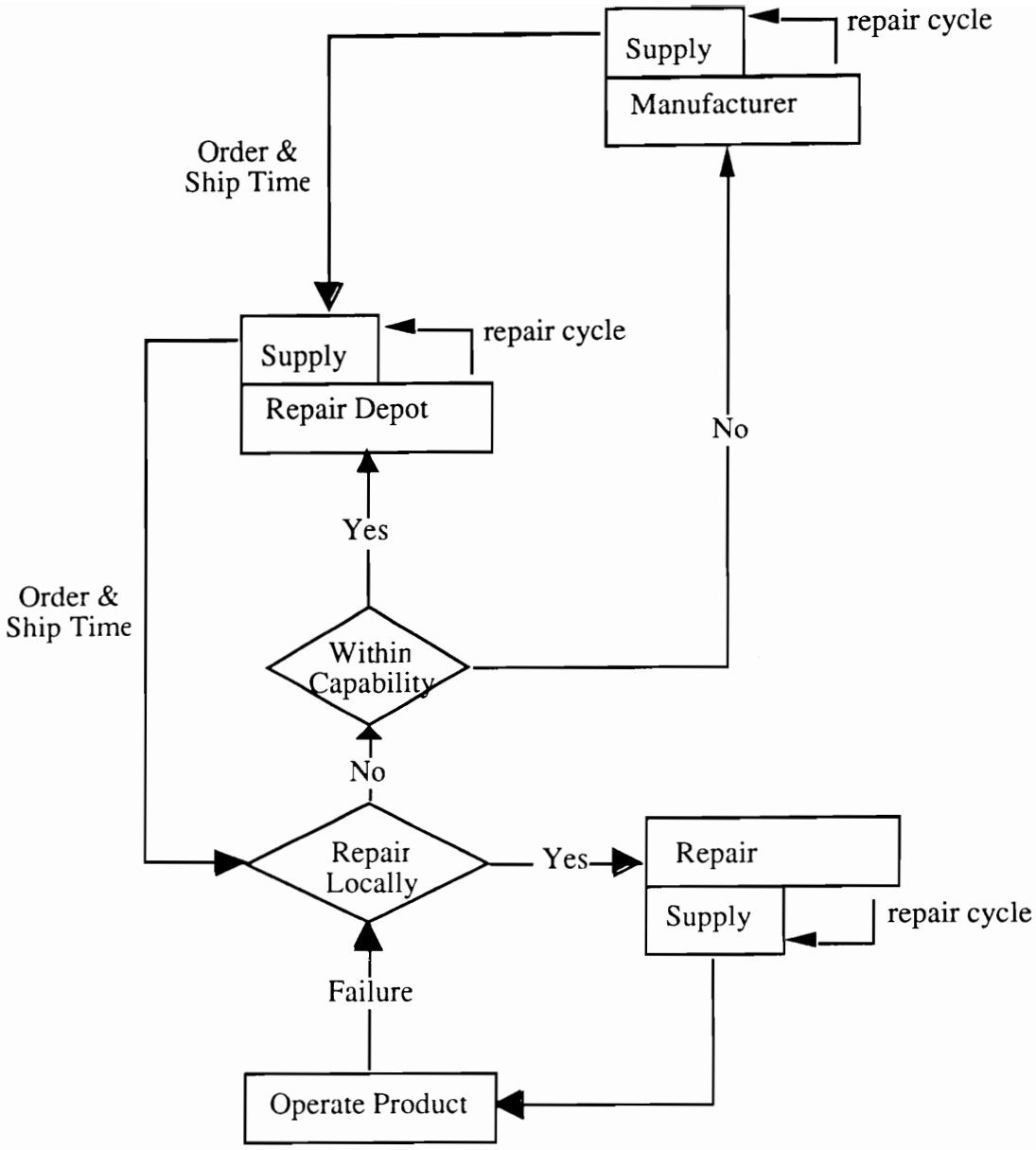


Figure 3.3. Support Structure

and computing the amount of spare assets to keep on-hand for the repair of incoming failed units. We assume static, system-myopic inventory policies and that once the product is fielded, the demand from the operating and repair locations for components and sub-assemblies will reflect steady state requirements of follow-on support to meet the daily operations.

The following list of assumptions will be used to govern the repair and inventory policies for each of the operating locations;

1. Components will be condemned automatically upon failure and repair will be attempted on all LRU's
2. If an LRU is to be condemned, this fact will be discovered at the repair facility
3. Each operating location will stock LRU's
4. Each repair location will stock components for each component type that goes into the LRUs that the repair location maintains
5. Repair facilities order components directly from the manufacturer
6. New LRUs are ordered by the repair facility which condemned them from the manufacturer on a one for one basis
7. The operating location orders replacement LRUs from the repair facility on a one for one replacement basis

The following list represents the decision variables for production control;

q_j = production batch size for component type j

sl_{jn} = safety stock held at location n for component type j

The final set of decision variables are those for field operations. If it is determined that repair should take place, maintenance policies, for scheduled and unscheduled repairs of the LRUs must be determined. These decisions involve determining where the repair action should take place. These policies will then influence decisions about stocking spare assets at each repair facility.

The following list represents the field operation decision variables;

$o1_{jn}$ = order quantity for component type j at location n

$o2_{\ell n}$ = base stock level for LRU ℓ at location n

The above stockage variables would apply to the depot repair facilities as well as the operating locations. If the operating location is designated as the repair site for a particular LRU, that location will need to stock the necessary components to replace the failed ones within the LRU. The location will also need to stock the LRUs so that a serviceable one is immediately available to replace the failed on while it is being repaired. The operating location must also stock those LRUs which are repaired elsewhere to replace the broken units while they are sent away for repair.

Model Parameters

Model parameters form the database that is required to evaluate the performance measures for given values of the decision variables. The parameters for this model formulation are separated into the categories of Product Design, Manugistics System Design and Operations Control which

are aligned with the phases of the product life cycle. The following notation and definitions are introduced for the set of parameters.

Product Design

λ_j = the failure rate for component type j

$\mu 1_{\ell n}$ = the mean time it takes to test LRU ℓ to determine which underlying component failed

$\mu 1_{j\ell}$ = the mean time to remove and replace component j from LRU ℓ

$\mu 2_{\ell n}$ = the mean time to remove and replace LRU ℓ at operating location n, before starting the repair action

[The set of repair locations indexed by n include the manufacturer's repair operation which is designated as location 0]

K_ℓ = mean condemnation fraction for LRU ℓ

$C1_j$ = cost of purchasing component j from an outside supplier

$C2_n$ = cost of repair labor per hour at location n

$J1_o$ = Indicates the configuration type for the set of requirements in configuration option o. It will take on a value of 1, 2 or 3 where; it equals 1 if the requirements are in a serial configuration, it equals 2 if the requirements are in an active redundant configuration, it equals 3 if the requirements are in a passive redundant configuration.

$J2_o$ = Ordered set of requirements which are contained in configuration option o. For the parallel case, the set will consist of one requirement listed k times where k is the number of times the requirement is in parallel with itself. For the passive redundant case, it is assumed the set of requirements is ordered to indicate the sequence in which stand-by units take over operation upon failure of the main unit and the other stand-by units.

$J3_r$ = Set of configuration options that are available to satisfy requirement r

$J4$ = Set of all sub-assembly *requirements* that represent the smallest units that would be removed and replaced to initiate a repair action. These sub-assemblies are called Line Replaceable Units (LRUs). The sub-assemblies within the set might consist of several requirements or just an individual requirement.

$J5_\ell$ = The set of feasible repair locations at which LRU ℓ can be repaired

$J6_\ell$ = The set of all component requirements within LRU ℓ

Manugistics System Design

$C3_p$ = cost of production labor per cell hour in cell p

$J7_j$ = set of all cells that can produce component type j

$W1_p$ = capacity of cell p in cell hours per year

$L1_p$ = existing production load on cell p from other components and products that are also being manufactured at the facility, in cell hours per year

$W2_n$ = capacity of repair location n in cell hours per year

$L2_n$ = existing repair load on location n from other product sub-assemblies that are also being repaired at the facility, in hours per year

$N1$ = number of cells in the current production system

ω_n = average wait time, in hours, for facility n

$J8$ = The set of all operating locations

$J9$ = The set of all repair locations. Since repair can also take place at the operating location, this set can also include operating locations.

Operations Control

$C4_{jn}$ = holding cost rate ($ per component-year) for component type j at location n

$C5_{\ell n}$ = holding cost rate ($ per LRU-year) for LRU ℓ at location n

$C6_{jp}$ = setup cost per setup for component type j in cell p

P_{jp} = mean cycle time for component type j in cell p

S_{jp} = mean set up time for component j in cell p

O_n = mean operating hours per year for the system at operating location n

O_r = mean operating time ratio for component requirement r; this is used if any particular module is operating differently from the product as a whole. For example, an aircraft will be pre-flight checked before take-off which causes certain modules to be operated before the actual flight. The operating time ratio for these modules could be set at a value of 1.2 to indicate that they operate 1.2 hours for every hour of operation of the product.

T_{nm} = mean transport time in hours between location n and m, the manufacturing location will be designated as location 0

$C7_{nm}$ = mean cost, per trip, from location n and to location m

t_i = time of acquisition of i^{th} final assembly, to allow for a phased-in deployment schedule

$N2$ = number of operating locations

$N3_n(t)$ = number of systems that are operated at location n at time t

$N4$ = number of years for system to operate

$N5$ = number of repair depots in the logistics system

$N6 = \sum_{t=1}^{N4} \sum_{n \in J8} N3_n(t)$ = total number of systems acquired at all operating locations

$C8_\ell$ = disposal cost for LRU ℓ

$C9_j(i)$ = average acquisition cost of component j for i^{th} final assembly

$C10_\ell(i)$ = cost to assemble LRU ℓ for i^{th} final assembly during acquisition

$C11_\ell(i)$ = cost to assemble LRU ℓ into final assembly during acquisition for i^{th} final assembly

Performance Measures

Performance measures provide us with the information that is needed to evaluate the operation of the system. For this dissertation, we have determined that the two most relevant performance measures are *LCC* and availability of the system. Several intermediate measures have been indentified to facilitate the calculation of the two main performance measures. While many of these measures can be categorized by the phases of the product life cycle, there is a great deal of interaction among them; consequently, they will be presented in the order of dependence. The following intermediate measures are introduced;

Average component demand rate at operating location n, from LRU ℓ

$$D_{j\ell n} = (\lambda_j)(O_n)(N3_n) \sum_{r \in J6_\ell} \delta(x_r - j)(O_r) \tag{3.1}$$

This measure computes the average demand rate for each component that is in LRU ℓ at every operating location. The delta function counts the number of times that component j is present in LRU ℓ. Every occurance of component j is weighted by its operating time ratio, O_r, which is then summed over all the requirements contained in LRU ℓ. This summation is multiplied by the number of units which are operating at location n, $N3_n$, the total operating time for each unit, O_n, and the failure rate of component j, λ_j.

Average demand rate for component j at repair facility n

$$D_{jn} = \sum_{r \in J4} \sum_{k \in J8} D_{j x_r k} \, \delta(m_{x_r k} - n) \tag{3.2}$$

Since the repair locations place the orders for additional component parts, it is necessary to know what the demand will be for these parts. This measure sums the component demands by LRU, from each operating location which is assigned to it for repair.

Average demand rate for component j at the manufacturing location

$$D_j = \sum_{r \in J4} \sum_{k \in J8} D_{j x_r k} \tag{3.3}$$

This measure computes the total demand that will be placed upon the manufacturing location for component j. The component demand rate by LRU is summed over all of the operating locations.

Average demand rate for LRU ℓ at operating location n

$$D_{\ell n} = (\lambda_\ell)(O_n)(N3_n) \tag{3.4}$$

The failure rate for the LRU, λ_ℓ, is computed from the component failure rates in a sequential fashion depending on the configuration of the components. This computation will be discussed in greater detail later. LRUs must be stocked at both the operating location and the repair location to minimize product down time. This measure is necessary for determining the base-stock levels for each LRU at the operating location.

Average demand rate for LRU ℓ at repair location n

$$D_{\ell n} = \sum_{k \in J8} D_{\ell k} \, \delta(m_{\ell k} - n) \tag{3.5}$$

The measure computes the demand rate for each LRU that is assigned to repair location n. The delta function determines which operating locations are assigned to repair location n. If an operating location is assigned to repair location n, then its LRU demand is added into the total demand for operating location n.

Average demand rate for LRU ℓ at the manufacturing location

$$D_{\ell} = \sum_{n \in J9} (D_{\ell n})(K_{\ell}) \tag{3.6}$$

Since LRUs are repairable, the manufacturer will only need to replace those LRUs which are completely worn out or destroyed beyond repair. The number of LRUs which must be replaced by the manufacturer is computed by taking the average LRU demand rate at the repair locations and multiplying it by the condmenation fraction, K_{ℓ}.

Average load for cell p at the manufacturing location

$$PL_p = LI_p + \sum_{s \in J4} \sum_{r \in J6_{x_s}} \delta(c_{x_r} - p)(D_{x_r})[P_{x_r p} + \frac{S_{x_r p}}{q_{x_r}}] \tag{3.7}$$

The new load for cell p is computed by taking the load for each component j that is assigned to cell p and adding it to the existing load. The load for each component consists of the average run time and the average setup time.

Average load for repair location n

$$RL_n = L2_n + \sum_{s \in J4} D_{x_s n} [\mu 1_{x_s n} + \sum_{r \in J6_{x_s}} (\mu 1_{x_r x_s})(D_{x_r x_s n})] \quad (3.8)$$

The new load for repair location n is computed by adding the load for each LRU that is assigned to be repaired at location n to the existing load. The repair load for LRU ℓ consists of the time to test the LRU to determine which components have failed, and the time to remove and replace the failed components.

Mean Manufacturing Leadtime =

$$ML_j = f(q_j, S_{jc_j}, P_{jc_j}, PL_{c_j}) = S_{jc_j} + (q_j)P_{jc_j} + g(PL_{c_j}) \quad (3.9)$$

The measure ML_j includes the expected time to produce a batch of size q_j for component j in cell c_j plus the average queue time in the cell. It is increasing in the production load at the cell where component j is produced. It is directly proportional to the setup time, the cycle time and the average wait time for the cell in which component j is produced. For now, the choice of queueing model for the function $g(PL_{c_j})$, which represents average wait time, will not be specified.

In order to determine the safety stock level for component j, we must have a criterion which measures the amount of protection we receive for a particular level of safety stock. We will use facility resupply time (ST_{jn}) as the criterion.

Resupply Time is the amount of time that is needed to obtain a serviceable spare asset from a higher echelon which may include repair time, transportation time and manufacturing leadtime.

There are several scenarios which are combined to compute a weighted average for resupply time. These scenarios depend on the designated level of repair and are affected by the out-of-stock conditions at each echelon.

Average Supply Time for component j at repair location n =

$$ST_{jn} = f(o1_{jn}, s1_{jn}; D_{jn}, T_{0n} + ST_{j0}) \tag{3.10}$$

where ST_{j0} is the average supply time at the manufacturing location and

$$ST_{j0} = f(q_j, s1_{j0}; D_j, ML_j)$$

Average Supply Time for LRU ℓ at operating location n =

$$ST_{\ell n} = f(o2_{\ell n}; D_{\ell n}, T_{m_{\ell n} n} + ST_{\ell m_{\ell n}}) \tag{3.11}$$

Mean Corrective Maintenance Time =

$$MCT_{jn} = \mu 1_{jn} + ST_{jn} \quad \text{for component } j \tag{3.12}$$

$$MCT_{\ell n} = \mu 1_{\ell n} + \frac{\sum_{r \in Y_r} (\lambda_{x_r})(w_r)(MCT_{x_r n})}{\sum_{r \in Y_r} (\lambda_{x_r})(w_r)} \quad \text{for LRU } \ell \tag{3.13}$$

When an LRU fails, testing is done to determine which underlying component caused the failure. Once the failed component has been identified, corrective action is taken to return the LRU to serviceable condition. The MCT of an LRU, then, is the weighted average of the component MCT's which make up the LRU, where the weights correspond to the failure rates (λ_i) and percent operating time $(w_r = \frac{O_r}{O_n})$ of the components. Percent operating time only becomes important for the passive redundant case because the backup systems do not operate until the prime system has failed.

The MCT of an LRU is summed over the individual component MCT's which make up the LRU. It is shown in equation (3.13) as the weighted average of the components contribution to maintenance time with weights based on their failure rates, λ_r, and percent operating time, w_r. When the LRU consists of a series or active redundant configuration, all components operate 100% of the time, so w_i is set to 1. For the passive redundant configuration, the weights w_i must be adjusted for the percentage of time that the prime system and the backup systems operate.

Total repair time for an LRU consists of test time, Mean Corrective Maintenance Time (MCT) and wait time. Test time is the amount of time spent determining which component failed while MCT is the amount of time spent replacing the failed unit. Wait time is the amount of time spent waiting for equipment or personnel to perform the repair action. Since we are assuming that a family of LRU's will be repaired at a given location, or a given station within a repair location, an average facility wait time will be applied to all members of that family.

Average Supply Time for LRU ℓ at repair location n =

$$ST_{\ell n} = f(o2_{\ell n}; D_{\ell n}, MCT_{\ell n} + g(RL_n), T_{0n} + ST_{\ell 0}, K_\ell) \qquad (3.14)$$

The depot may serve as an intermediate stocking location and the operating location can call upon the depot to provide spares support. In this case, resupply time will contain the additional time it takes to transport the requested spare asset. At any repair location, if the spare asset is unavailable at the repair facility, that facility can request additional spares from the manufacturer. In this case transportation time will once again be added to resupply time. However, if the manufacturer does not have the requested assets, a production run may be scheduled and the

manufacturing leadtime will be added to the resupply time. Each of these scenarios is averaged together to compute resupply time.

If the decision is to repair at the depot, then the operating location should stock enough spares to cover the amount of time it takes to ship the broken units through the repair pipeline. The only difference between this scenario and the ones outlined above is that the operating location will not stock individual components to support repairs. The last scenario is where repair occurs at the manufacturer. In this case, the depot may stock LRUs but not components.

Mean Down Time =

$$MDT_{\ell n} = f(ST_{\ell n}, \mu 2_{\ell n}) = \mu 2_{\ell n} + ST_{\ell n} \quad (3.15)$$

$MDT_{\ell n}$ is the average time that a LRU ℓ is unavailable to the system at an operating location due to the time required to remove and replace the failed LRU and the time required to receive a working LRU from the supply system. If a spare LRU is not available at the operating location, the system must wait through the leadtime that is needed to obtain an operational spare from a repair site. It will be assumed each repair facility is dedicated to a specific 'family' of parts. For example, the group of sub-assemblies belonging to an engine assembly requires unique equipment or technicians and so will be maintained at a location different from an electronics sub-assembly.

The expected time for an LRU's repair cycle is a function of the expected resupply time at the repair site and the expected time to complete the remove/replace action. The probability of the system having to wait for a spare LRU to be delivered from a higher echelon depends on the

safety stocks of LRU ℓ through the supply chain of LRU ℓ and the demand for spares determined by the failure rates of the components in the LRU. MDT is decreasing in the safety stocks, increasing in the failure rates, repair times and transportation times.

For a disposable item, if the item is out of stock at the operating location, then the depot will provide the spare. If the depot is out of stock, then the manufacturer will provide the spare. If the manufacturer is out of stock, then mean down time is extended by the average remaining manufacturing leadtime. For a repairable item, if the operating location is out of stock then the leadtime will also include the amount of time to repair the item, unless it has been condemned, if the depot or manufacturer does not have a replacement asset. The computation of replacement spares will take into consideration the condemnation rate of each LRU.

The failure rate of an LRU, λ_ℓ, can be built up from the component failure rates, λ_j, which make up the LRU. The manner in which the failure rates of the components are combined depends on the configuration of the components within the LRU. Each component contributes to the failure rate of the LRU by either increasing it or decreasing it depending on whether the component is in series with other components or is redundant to the other components. The following recursive relationship holds for computing LRU failure rate;

$$\lambda_l = \lambda_l + h\frac{1}{MTBF_o}$$

where the subscript o represents the underlying configuration options of the LRU and may be a sub-assembly or a component. The constant h takes of different values depending on the

configuration of the sub-assemblies and components. This formula will be discussed in greater detail later in the dissertation.

LRU Availability at operating location n =

$$A_{ln} = f(Y_l;\ \{\lambda_{x_r}: r \in J6_l\},\ MDT_{ln}) \tag{3.16}$$

System Availability at operating location n =

$$A_{sn} = f(\{A_{x_r n}: r \in J4\}) \tag{3.17}$$

Fleet-wide System Availability across all operating locations =

$$A_s = f(\{A_{sn}: n \in J8\}) \tag{3.18}$$

This measure computes the probability of the system being in an operational state upon demand. LRU availability is a function of Mean Down Time, discussed in the previous section, and the system-level Mean Time Between Failure (MTBF) which is discussed below. The marginal behavior of the availability function increases as the system reliability increases or the expected mean down time decreases.

When a system of modules[5] is in a series configuration, the system fails when any of the modules fail. We will assume that the operation of the system stops during replacement of the failed module. Under this assumption, Barlow and Proschan [1981] offer a theorem that states, the MTBF of a system of N elements in series approaches $(\sum_{i=1}^{N} \lambda_i)^{-1}$, where λ_i is the failure rate of the ith component, over time, regardless of the failure distribution of the elements. Since the

[5]The word 'component' will be used when referring to an individual component type in the product structure, while 'module' will be used when referring to any sub-assembly of component types.

logistics engineering problem that is the subject of this dissertation measures average availability over the lifetime of the product, this assumption will hold. Consequently, when an LRU is represented by modules in a series configuration, it is easier to compute its MTBF using the modules' failure rates, where failure rate is the reciprocal of MTBF. In the series case, the LRU failure rate is the sum of the modules' failure rates.

When an LRU is represented by modules in an active or passive redundant configuration, it is easier to compute its MTBF using the modules' MTBFs. In the active redundant case, it will be assumed that the same module type is used to fill the parallel requirements. Barlow and Proshcan [1981] state that LRU MTBF is then equal to the component MTBF multiplied by $\sum_{i=1}^{p} \frac{1}{i}$, where p is the number of components in parallel. For the passive redundant case, the LRU MTBF is equal to the sum of the modules' MTBFs.

It will be assumed that each component has exponentially distributed failure time with a cumulative distribution function, $F(x_j)$, and that failures of components and sub-assemblies occur independently. When a component has an exponential failure distribution, scheduled maintenance does not add to the life of the component. For this reason, MDT is assumed to be a function of unscheduled maintenance alone. It will also be assumed that repair of a sub-assembly returns it to 'as good as new' condition.

The embedded availability calculations are based on standard formulations and are a function of the product configuration, component failure rates and MDT. The individual component failure rates are combined to form sub-system failure rates which are also combined to determine the

overall system failure rate. MDT is computed by configuration option and is built up from individual component requirements as described above.

Using Figure 3.1 as an example of a product structure and assuming that requirements A, B and C represent LRU's, we can calculate the availability of requirement C as follows; if component 6 is chosen to fill requirement I, then the MTBF for configuration option 4 is calculated using the failure rates for component 6 in the following manner;

$$MTBF_4 = \sum_{i=1}^{2} \frac{1}{i} MTBF_6 = \frac{3}{2} MTBF_6$$

and,

$$A_{4n} = \frac{MTBF_4}{MTBF_4 + MDT_{4n}}$$

where MDT_{4n} is computed using equation (3.15).

If component 8 is chosen to fill requirement J and component 10 is chosen to fill requirment L, then the availability for configuration option 5 is computed in the following manner;

$$MTBF_5 = \sum_{r \in J3_5} MTBF_r = MTBF_8 + MTBF_{10}$$

The MTBF's are additive because the standby system is not functioning until the primary system fails. The MDT is computed as outlined in equation (3.15). The availability for option 5 is;

$$A_{5n} = \frac{MTBF_5}{MTBF_5 + MDT_{5n}}$$

Finally, if option 4 is chosen to fill requirement D and option 2 is chosen to fill requirement C, then the availability of requirement C is;

$$\prod_{r \in J2_2} A_r = (A_D)(A_E)$$

We can calculate A_{Sn} from the availabilities of the LRU's as in equation (3.17) depending on the configuration of the LRU's within the system. For example, if N LRU's are arranged in a series configuration, A_{Sn} is simply the product of the availabilities of the N modules, or $A_{sn} = \prod_{t=1}^{N} A_{tn}$. If N LRU's are arranged in a parallel configuration, then availability is calculated from the complement of the LRU availabilities as, $A_{sn} = 1 - \prod_{t=1}^{N}(1-A_{tn})$. If the modules are in a standby configuration, then calculating availability is not straight forward but can be accomplished using Markov analysis [Lewis, 1987].

The following equations represent the general formulas that will be used to compute the system level availabity performance measure;

Operational System Availability at operating location n = $A_{sn} = \dfrac{MTBF_{sn}}{MTBF_{sn} + MDT_{sn}}$ (3.19)

Overall fleetwide average availability = $A_s = \dfrac{\sum_{n=1}^{N1} A_{sn} * N2_n}{\sum_{n=1}^{N1} N2_n}$ (3.20)

The LRU's in the product structure represent a dividing line in the computation of system availability. Working upward from the bottom of the product structure, we combine the MTBF's, the Resupply Times, the production and repair loads to build a MTBF and MDT for the LRU. Once these measure have been calculated for the LRU's in the product, we no longer use the component measures, instead we use the LRU measures to compute availability for the system. Figure 3.4 illustrates the heirarchical relationship among these performance measure and also how they are combined to formulation LRU and systemwide availability.

Manugistics System Design

The manufacturing facility is assumed to have a cellular arrangement for producing the components that are chosen to be built internally. It is also assumed that the facility is producing other components and this new design will be adding a certain load to the existing cells. Cell assignment is based on the cost of producing component j in cell p without violating the capacity limit of the cell.

The production system costs are built from a series of computations which start with the expected demand for the components and sub-systems. Expected demand is a linear function of the component and sub-system failure rates and is equivalent to the expected number of failures per unit time. Once the expected demand has been determined, the production batch size and the number of production batches per unit time can be calculated. The average number of production batches per unit time is equal to the expected demand per unit time divided by the batch size. Similar to the MTBF calculations, total LCC can also be calculated in a series of stages based

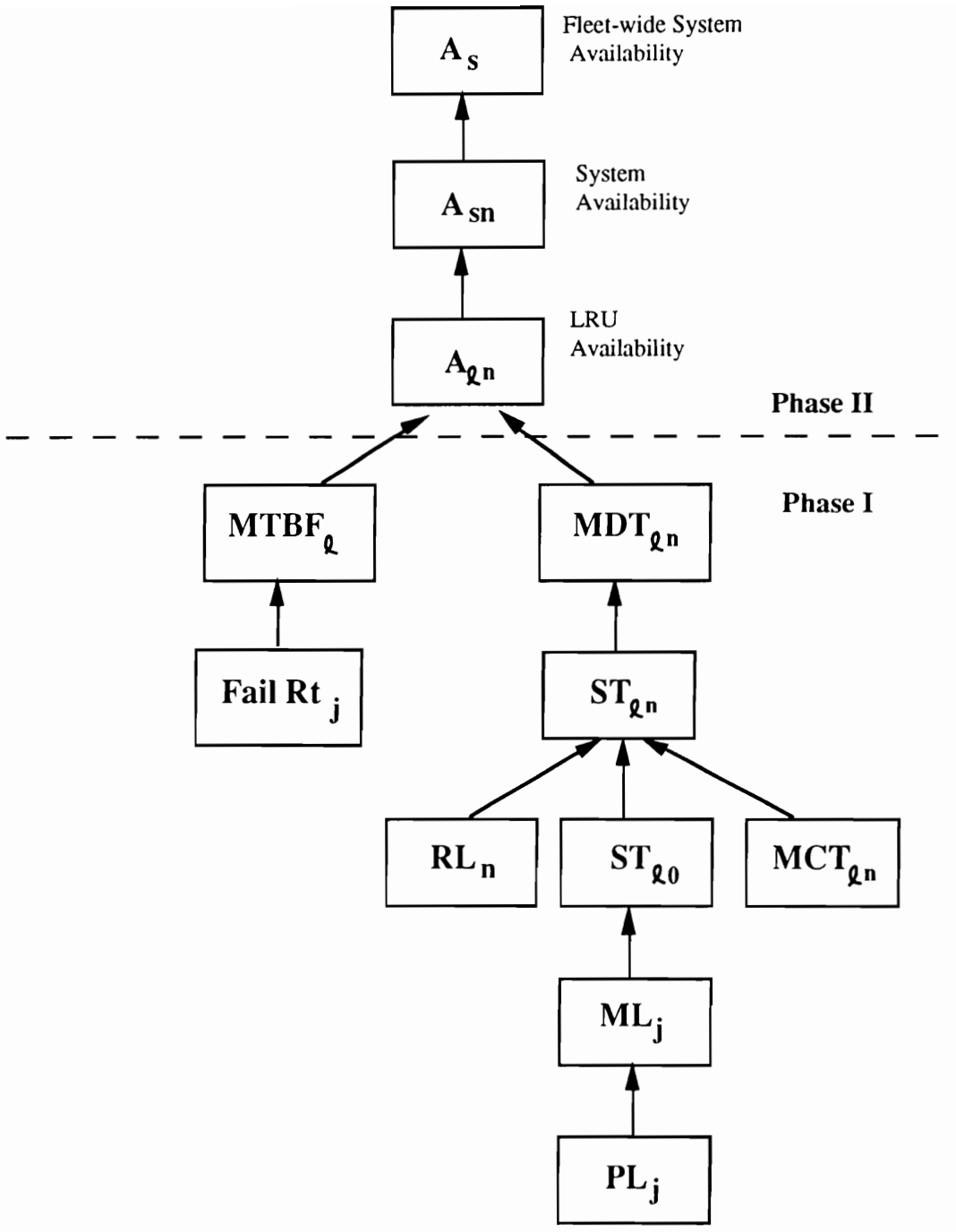


Figure 3.4. Performance Measure Hierarchy

on the contribution to cost of each sub-assembly in the product structure. The following cost measures are introduced for the Life Cycle Cost computations;

Number of production cycles per year for component type j = $\frac{D_j}{q_j} = N_{jp}$

Average Production Cost per unit time =

$$PC = \sum_{s \in J4} \sum_{r \in J6_{x_s}} \sum_{p=1}^{NI} \{[D_{x_r} P_{x_r p} C3_p + \frac{D_{x_r}}{q_{x_r}} C6_{x_r p}] \delta(c_{x_r} - p)\} \quad (3.21)$$

Average production cost is a function of the number of production runs, the cell to which component *j* has been assigned and its associated production and setup costs. If demand remains fixed, production cost decreases as batch size increases because this reduces the number of production cycles and also the average number of setups to complete the production run.

Average Repair Cost per unit time at location *n* =

$$RC = \sum_{n \in J9} \sum_{s \in J4} C_{2n} D_{x_s} [\mu 1_{x_s n} + \frac{\sum_{r \in Y_s} \lambda_{x_r} w_r \mu 1_{x_r n}}{\sum_{r \in Y_s} \lambda_{x_r} w_r}] \quad (3.22)$$

The construction of the average repair cost is similar to the derivation of MCT_{t_n} for the LRU's repaired at location *n*. Repair cost is calculated from the number of failures for each LRU, the expected length of time needed to perform the repair action, the labor cost of the repair facility and the specific repair location that is chosen. It is dependent on the product configuration and connectivity between the sub-assemblies. Certain configurations are easier to maintain because of accessibility to the sub-assemblies and components. Repair cost is an increasing function in the expected number of failures and the expected repair time.

Operations Control

The next set of performance measures describes the decisions that are made during the daily operations of the production facility and the operating locations. These decisions involve determining the production batch sizes, inspection policies and sample sizes, level of repair and spare asset quantities at each location.

Since it is assumed that the manufacturer can also serve as a repair site for the product, he must carry inventory to support repair actions as needed. The inventory will consist of both lower level components and fully built LRUs. The replacement LRU will be shipped to the location which experience the failure to replenish their serviceable asset pool and the components must be available for the repairs actions against the LRUs. This will also hold true for the operating locations and any intermediate repair facilities.

Two different batch size computations are needed to support the decision making process at the production facility and operating locations. The performance measures for all inventory models are **Average Annual Setup** and **Ordering cost**, **Holding cost** and **Service Level**. The cost measures are combined to compute a total cost which is a non-linear convex function of **Production Batch Size** and the expected annual demand of each sub-assembly and component.

Average Inventory Cost per unit time at the manufacturing location =

$$ICI = \sum_{s \in J4} \sum_{r \in J6_{x_s}} f(q_{x_r}, sl_{x_r 0}, D_{x_r}, C4_{x_r 0}) \quad \text{[component]} \tag{3.23}$$

Average Field Inventory Cost per unit time at location n =

$$IC2_n = f(o1_{jn}, s1_{jn}; C4_{jn}, D_{jn}) + f(o2_{tn}; C5_{tn}, D_{tn}) \; n=1..N1 \tag{3.24}$$

This cost consists of the inventory of components that must be held if location n is a repair location and the inventory of LRU's that must be held to replace LRU's that have failed. The inventory of LRU's must cover the amount of time it takes for one repair cycle of the LRU as well as include additional units to replace LRU's that have been condemned.

Average Field Inventory Cost for all locations =

$$IC2 = \sum_{n \in J9} IC2_n + \sum_{n \in J8} IC2_n \tag{3.25}$$

Average Systemwide Transportation Cost =

$$TC = \sum_{r \in J4} \sum_{n \in J8} 2C7_{n\, m_{x_r, n}} D_{x_r, n} + \sum_{s \in J4} \sum_{r \in J6_{x_s}} \sum_{n \in J9} C7_{0\, n} \frac{D_{j\, x_r}}{o1_{jx}} + \sum_{r \in J4} \sum_{n \in J9} C7_{0\, n} D_{x_r, n} K_{x_r} \tag{3.26}$$

Transportation cost is incurred when a failed LRU must be sent to an intermediate repair facility or the manufacturer for repair. It is assumed that the number of trips to the depot or warehouse would be equivalent to the expected number of failures at a given location since the failed unit would be immediately sent to the repair facility. This cost is a function of the transportation rate, the distance between the operating location and the repair facility and the expected number of failures. It is a linear function in these parameters and is directly proportional to them. The average cost of shipping components to the repair facilities from the manufacturer is based on the average number of batches of components sent per unit time. We assume each batch is ordered separately. The cost of shipping new LRU's to the repair facilities from the manufacturer depends on the condemnation rate of LRU's at the repair facilities.

Average Disposal Costs per unit time =

Average Disposal Cost of LRU's + Average Disposal Cost of components =

$$DC = \sum_{n \in J9} \sum_{s \in J4} [C8_{x_s} K_{x_s} D_{x_s n} + \sum_{r \in J6_{x_s}} C8_{x_r} D_{x_r n}] \quad (3.27)$$

Disposal cost is the annual expected cost of condemning the failed LRUs and components upon wearout. It is based on the expected number of failures, the disposal cost per unit and the condemnation rate.

Acquisition Cost =

$$AC(i) = \sum_{s \in J4} \sum_{r \in J6_{x_s}} [C9_{x_r}(i) + C10_{x_s}(i) + C11_{x_s}(i)] \quad (3.28)$$

This measure represents the cost of obtaining the i^{th} final assembly during the initial acquisition of the product for all locations.

Expected Provisioning/Procurement Cost =

$PPC(t) = f(\{N3_n(t): n \in J8\})$

This measure represents the present value of follow-on support and operating costs that are incurred once the product has been operating in the field. It is a function of the number of final assemblies that have been located at each operating location. The various support and operating costs that are contained in the provision/procurement cost are illustrated by the following formula;

$$PPC(t) = \beta'[PC(t) + RC(t) + IC1(t) + IC2(t) + TC(t) + DC(t)] \quad (3.29)$$

where β^t is a discount factor. The final product may operate over a period of several years so the discount factor will ensure that cost measures are all in net present dollars.

Total LCC Cost =

$$LCC = \sum_{t=1}^{N4} PPC(t)\beta^t + \sum_{i=1}^{N6} AC(i)\beta^{t_i} \qquad (3.30)$$

The total Life Cycle Cost for the product is the sum of the expected provisioning/procurement costs and the acquisiton costs. This overall cost and the systemwide availabilty are the two components of the objective function for the model.

Model Statement

The objective of this research is to develop a decision support model which will optimize system life cycle performance with respect to design variables. The model will evaluate alternative design solutions by calculating the associated operational availability as well as manufacturing and logistics support costs. The many performance measures are combined into the following model formulation;

Min Life Cycle Cost

Max System Availability

subject to

Product Design Requirements

The following outline presents the relationships among the models, and how they combine to form the above objective functions. We categorize the performance measures in a hierarchical scheme of three levels to illustrate the sequence of operations required to solve the overall problem.

The first level consists of embedded descriptive models which map the decision variables into the performance measures. For illustrative purposes in this dissertation, simple models were chosen with the assumption that other more complex models from the literature could be substituted while preserving the basic model characteristics such as, monotonicity or convexity and the sets of variables and parameters on which each performance measure depends. The second level of the hierarchy maps the performance measures into the major criteria of Life Cycle Cost and system availaiblity. The final level represents the multi-criteria function which combines the major criteria from the previous level. The levels are defined as follows;

Level 1;

Product Design Models → f(product design variables, product design parameters)

Manugistics System Design Models → f(manugistics design variables, manugistics design parameters)

Operations Control Models → f(operations control variables, operations control parameters)

where the variables and the parameters for each type of model were outlined above.

Level 2;

Life Cycle Cost Model → f(acquisition cost, operating cost, repair cost, production cost)

Availability Model → f(mean time between failure, mean down time)

Level 3;

Utility Model → f(LCC, Availability)

Figure 3.5 shows the structure of this hierarchy of performance measures and models.

Using operational availability and Life Cycle Cost (LCC) as the main objective functions for the system design, the model can be broken out into several stages involving the various models. The overall objective is to maximize system availability and minimize LCC. These two measures conflict and will need to be combined into one utility function to allow for trade offs to occur between them.

There is a natural separation of the variables to be optimized within this problem. They fall into the categories of Product Design variables, Manugistics System Design variables and Operations Control variables. These variables can be grouped together and solved as stages within the overall framework of the problem. The overall linking variable which connects each of the stages together is the choice of configuration option (x_r). This separation of the variables can be exploited while determining an optimal solution to the problem.

Decomposition techniques are generally used to solve large-scale or complex math programming problems. These methods can be applied to convert large problems into smaller problems that are more manageable, and can be coordinated to solve the larger problem. The Dantzig-Wolfe and Bender's algorithms make use of the decomposition principle to accomplish this task.

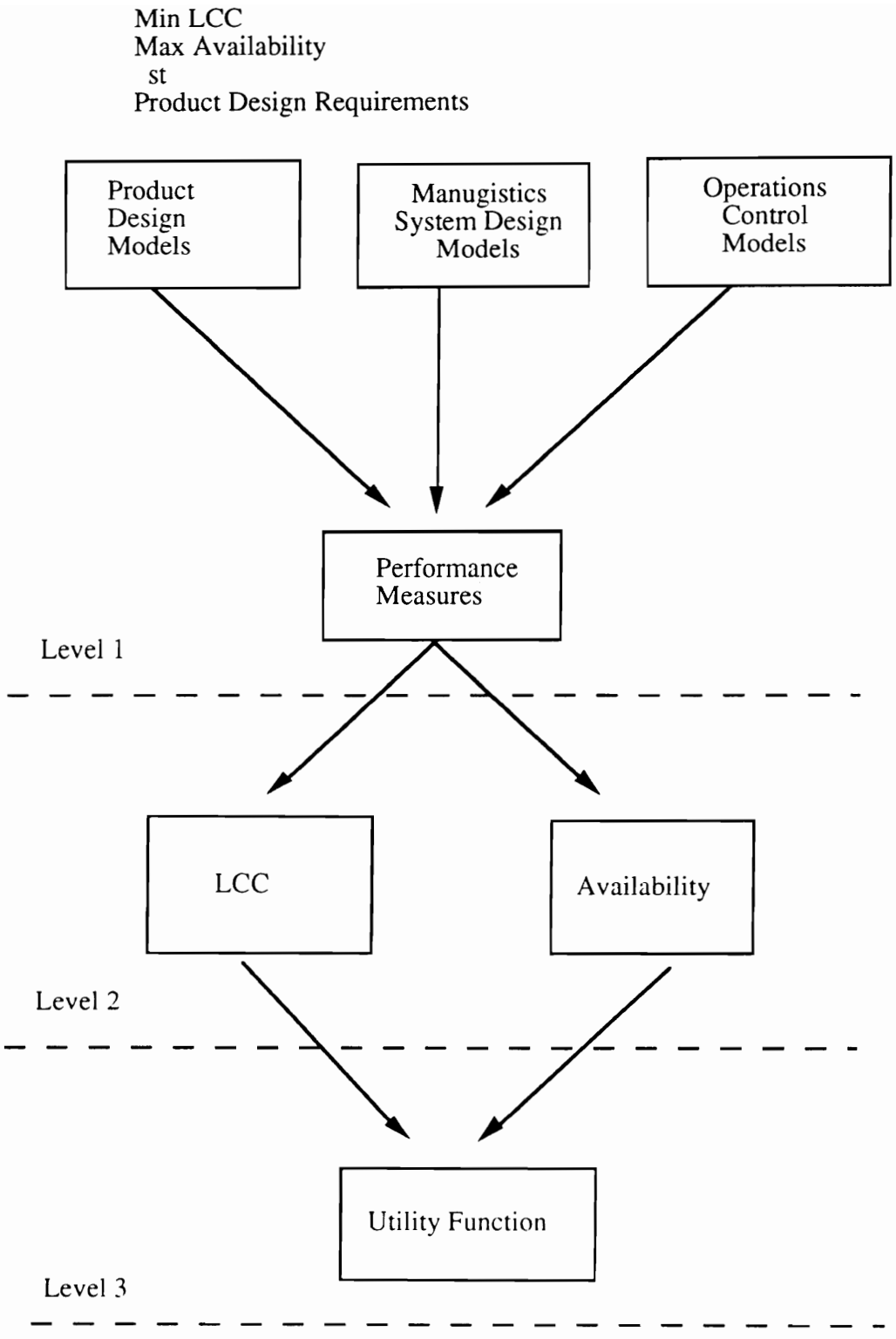


Figure 3.5. Model Hierarchy

The decomposition principle is a procedure for solving math programming models that offers the opportunity for separating the problem into smaller ones with special structures that can be solved more efficiently. The overall problem is optimized by passing information from the main problem to the set of smaller ones and back again. An iterative approach can be applied to optimize the smaller problems that are a part of the larger one. As the variables of one problem are optimized, their values are passed to the other problems to be used as inputs to determine their optimal solutions. These solutions are then passed to the first problem, and the technique is repeated until there is only a marginal improvement in the objective function value.

Chapter 4: Optimization Problem

A generic formulation of the problem and a representation of the solution was presented in Chapter 3 in terms of the variables to be optimized, the parameters to be included and the performance measures of interest. This chapter will expand on this model and discuss specific solution techniques that can generate an optimal solution to the problem.

General Problem Formulation

The goal of Concurrent Engineering is to ensure that the impact of design decisions are considered before beginning the production cycle. This dissertation specifically addresses the links among product configuration, component choice, and production costs, support costs, manugistics system design, operational control policies and system availability. The model will accept design information provided by the engineer as input and compute system level operational availability, manufacturing and logistics support costs for the expected lifetime of the system. The general formulation was fully described in the Model Formulation section of Chapter 3.

Decision Variables

The decision variable types are a mixture of integer and real types which contributes to the complexity of the problem. The product design variables represent the choices among alternate system configurations and specific component brand names. It takes on an integer value equal to the index number of the configuration option to be included in the design. The final product structure can be represented by the vector of design configuration choices, **x**.

The manugistics system decision variables dictate in which cell a component will be manufactured, the total number of cells required to build the product, the location at which maintenance should take place and the set of repair locations which can maintain a specific sub-assembly. All of these variables are integer types.

The operations control variables represent the production and logistic support decisions. They are the safety stock by location, the production batch size, and spares order quantities by repair site. These variables are based on annual average numbers and in reality have integer values, but we follow custom and treat them as real numbers.

Objective Function

The objective function contains two components that are conflicting in nature. The customer is interested in minimizing Life Cycle Cost (*LCC*) and also in maximizing system availability. Availability is increased when reliability increases or when system down time decreases. Highly

reliable components generally are more expensive to produce or purchase than less reliable components thereby requiring higher *LCC* for higher availability. System down time can be decreased by either improving repair turn time or maintaining higher levels of safety stock at each operating location which again increases *LCC*. The two criteria of minimizing *LCC* and unavailability are combined into a single objective function as $LCC + \gamma U$, where U is the system unavailability and γ is the marginal rate of substitution between *LCC* and U, subjectively specified by the decision maker. The decision maker will have to specify the tradeoff between *LCC* and U based on his belief as to the worth of each measure.

The model will compute optimal values for system *LCC* and U, given a specific value of the tradoff parameter γ. These optimal values are computed using the embedded solution techniques that are discussed later in this chapter.

This objective function can be separated into individual constituent functions which are combined in stages as each configuration option is considered in the product design. In what follows, we will develop an optimization procedure which takes advantage of the behavior and functional dependencies of these functions to break down this large-scale optimization problem into manageable sub-problems.

Methods for Solving Large-Scale Problems

Geoffrion [1970] presents a classification scheme for the types of large scale math programming problems and their associated solution techniques. He divides the optimization methodologies

into two high-level categories; problem manipulations and solution strategies. Problem manipulations translate a given problem into an alternative formulation that can be more easily solved while solution strategies reduce a problem into a related sequence of simpler optimization problems. Dualization of a linear program is an example of a problem manipulation and the Feasible Directions technique is a solution strategy. Most optimization techniques can be derived from various combiniations of these manipulations and strategies.

Geoffrion also states that problem size is not the sole factor in determining whether or not a problem should be classified as 'large scale'. Quite often, large scale problems have a special structure that can be exploited by the two classes of solution techniques mentioned above. Some of the more common structures include; multidivisional, combinatorial, dynamic and stochastic. Multidivisional problems contain interrelated 'subsystems' like divisions within a company or modules of a product. Combinatorial problems have a large number of variables which represent many possible options, like routes, schedules or component choices. Dynamic problems allow for changing conditions over many time periods and stochastic problems include uncertanty. The problem discussed in Chapter 3 can be classified as a large scale problem because of its mutlidivisional characteristics. It has several natural separations that can be exploited by manipulation techniques to make it more manageable to solve. The specific techniques and solution strategies that will be applied to this problem are discussed in the following section.

Partitioning and Dynamic Programming

One purpose of problem manipulation is to find an alternative formulation which more easily allows existing optimization algorithms to be applied in order to generate a solution. We do this by taking advantage of any special structures that are displayed by the original problem. Problem manipulations also look for any separability in the original problem which can be solved as independent sub-problems.

Partitioning, also known as 'projection', is one type of problem manipulation which takes advantage of the new problem structure that is produced when certain variables of the original problem are temporarily fixed in value. Partitioning can be applied in a sequential manner by first projecting onto a subset of variables, and then onto a subset of those variables in a sequential fashion. The new formulations can be viewed as a dynamic-programming reformulation which can then be solved in stages.

If the original problem is of the form;

$$Min_{d_1 \in D1,\, d_2 \in D2}\ f(d_1,d_2)\ \ st\ \ g(d_1,d_2) \geq 0 \qquad (4.1)$$

where $\mathbf{d}_1$ and $\mathbf{d}_2$ are two vectors representing a partitioning of the decision variables. The problem's projection onto the space of the $\mathbf{d}_2$ variables alone is;

$$Min_{d_2 \in D2}\ [inf_{d_1 \in D1}\ f(d_1,d_2)\ st\ g(d_1,d_2) \geq 0] \qquad (4.2)$$

which is an equivalent form of the above equation. The 'inner' optimization problem is evaluated for fixed values of the d_2 variables which reduces the feasible region to be evaluated. Formulation (4.2) becomes valuable for both the inner and outer problem when the new problem

formulations are easier to solve than the original problem. No restrictions are placed on the behavior of the functions f and g and the initial problem of (4.1) does not have to be of any special structure.

Geoffrion [1970] offers the following theorem;

Theorem 1. Problem (4.1) is infeasible or has unbounded value if and only if the same is true of problem (4.2). If (d_1^0, d_2^0) is optimal in (4.1), then d_2^0 must be optimal in (4.2). If d_2^0 is optimal in (4.2) and d_1^0 achieves the infimum of $f(d_1, d_2^0)$ subject to $d_1 \in D1$ and $g(d_1, d_2^0) \geq 0$, then d_1^0 together with d_2^0 is optimal in (4.1).

The projection technique can be generalized to n stages with the following formulation;

$$Min\ f(d_1, \ldots, d_n) \tag{4.3}$$

where the d_i (i = 1..n) are vectors representing a partitioning of the decision variables. Then an equivalent formulation of equation (4.3) would be;

$$Min_{dn \in Dn} [Inf_{dn-1 \in Dn-1} [Inf_{dn-2 \in Dn-2} \ldots [Inf_{d1 \in D1} f(d_1, \ldots, d_n)] \ldots] \tag{4.4}$$

which creates a sequence of sub-problems in place of the single problem shown in equation (4.3). Each of these sub-problems can represent a stage of a dynamic programming problem.

Mitten [1963] states that dynamic programming is a method for converting large-scale optimization problems into a series of equivalent problems, or stages, that contain fewer decision variables and, hence, are easier to solve. Solving the resulting smaller sub-problems will then lead to the overall optimal solution to the original problem. At each stage, the state variables represent all of the information about the variables of the outer problems that are relevant to the

decisions made at that stage.

Each individual stage within a dynamic programming problem has the following characteristics which are outlined below;

1. The input state variables at stage k, z_k, will depend on either the decision made at previous stages or other external conditions
2. The decisions, d_k, made at each stage are combined to form the overall optimal solution
3. The output state variables of one stage serve as the input state variables to the next stage and are built through a transformation of the input and decision variables. This can be represented by the notation, $z_{k-1} = T_k\ (z_k,\ d_k \in D_k)$, where T_k is the transformation function.
4. The return function, g_k, measures the stage's contribution to the problem's objective function. This can be represented by the notation, $g_k(z_k,\ d_k \in D_k)$

Figure 4.1 shows a typical stage within a dynamic programming process. When the stages are linked together in the fashion shown in Figure 4.2, the problem becomes a multi-stage decision process. The output of a stage, as described above, also serves as the input to the next stage within the multi-stage process. A recursive relationship of the stage returns can be identified as follows;

$$R_k\ (z_k,\ d_k \in D_k) = g_k\ (z_k,\ d_k \in D_k) + f_{k-1}\ (z_{k-1}) \tag{4.5}$$

where

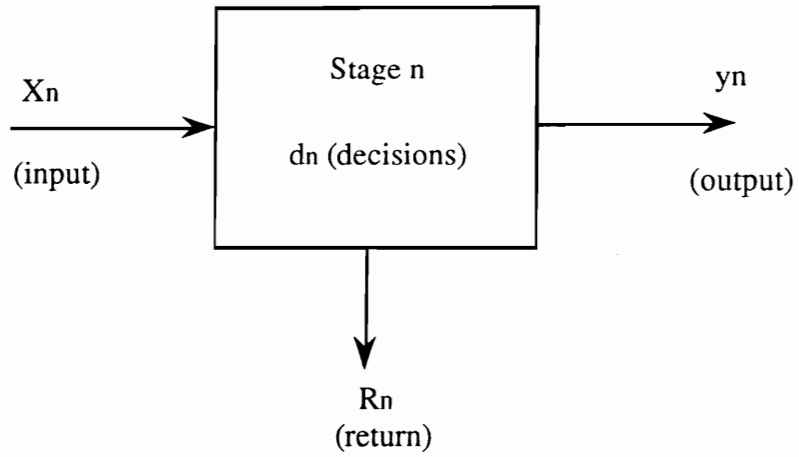


Figure 4.1 Typical stage (n) of a multistage process

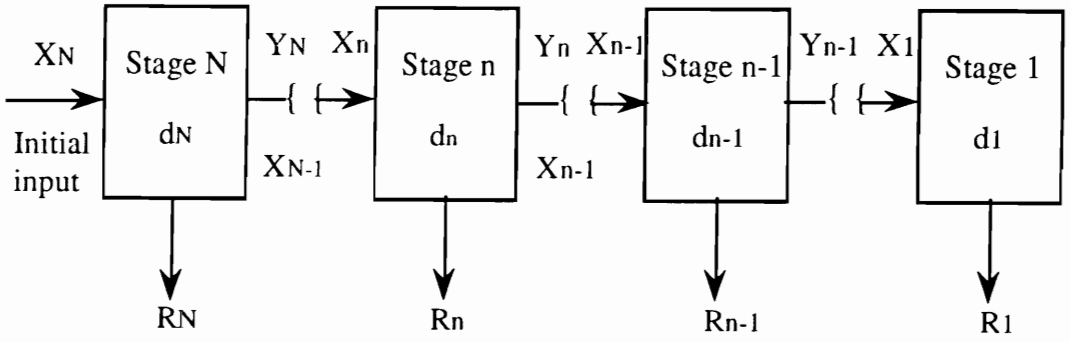


Figure 4.2 Simple multistage process

$$z_{k-1} = T_k\ (z_k,\ d_k \in D_k)$$

and

$$f_k\ (z_k) = Min_{d_k \in D_k}\ [R_k\ (z_k,\ d_k \in D_k)]\ for\ n = 1..N$$

over all feasible states z_k. The quantity, $R_k\ (z_k,\ d_k \in D_k)$ is the combined return through stage k and $f_{k-1}\ (z_{k-1})$ is the optimal return through stage $k-1$.

The principle of optimality says that, given the current state, the optimal decisions of the remaining stages must not depend on previously reached states or previous decisions. This means that the optimal policy for the N-stage problem, $d_N^* \in D_N\ (z_N),\ \ldots\ ,\ d_1^* \in D_1\ (z_1)$ must be such that any subset of decisions, $d_k^* \in D_k\ (z_n),\ \ldots\ ,\ d_1^* \in D_1\ (z_1)\ for\ n = 1..N$, is optimal for the last k stages for any input z_k. The input, z_k, is treated as a parameter and so we must determine the optimal value of the decision variable, d_k, for every feasible value of z_k.

The state z_k is, in general, a vector consisting of several components. The computational effort required to solve dynamic programming problems increases not only with the number of stages but also with the dimensionality of state variables as well. Accordingly, one should reduce the dimension of the state variables whenever possible.

Application of Dynamic Programming

Within any dynamic programming problem, the state variables, the decision variables, the transformation function and the stages must be identified. In defining the stages, we look to the

problem structure for any separations where it would be possible to identify a vector of state variables with low dimensionality. This means that the definition of stages is accomplished by looking ahead to the issue of determining state variables and transition laws.

For the problem outlined in Chapter 3, there is a natural separation of the performance measures along three sets of variables. We will call S_1 the set of operations control variables, S_2 the set of manugistics system design variables and S_3 the set of product design variables. These sets of variables will be represented as three major stages in the dynamic programming problem as follows;

$$Min_{S_1}\,[Inf_{S_2}\,[Inf_{S_3}\,f(S_1,\,S_2,\,S_3)]] \tag{4.6}$$

where $f = LCC + \gamma\, U$. It is a fortunate coincidence that the mathematical requirements for the most efficient sequence of projected problems produces the sequence shown in (4.6) which parallels the traditional decision-making phases of product design, manugistics system design and operations control.

The three major stages described above provide the order in which the decision variables are to be optimized. As we progress from the outer to the inner problem, the decision variables from the outer stages define the "system state" for the inner stage. The manugistics design variables, which determine a feasible set of repair and manufacturing locations to be considered, and the operations control variables of safety stock and batch size are combined to specify the state information for the innermost optimization over the product design variables. The effect of safety stock and batch size on the product design choices can be summarized by another variable, mean resupply time.

We define Mean Resupply Time (ST) as the average time required to return a serviceable item from a stocking location. Mean Resupply Time is a function of safety stock, order quantity and the distribution of demand during the leadtime. Each repair location will have a Mean Resupply Time. Inventory models used by the Air Force [AFLCR 57-4] and other military services, set optimal or near optimal safety stocks and order quantities uniquely once desired Mean Resupply Times are set. Therefore, in a problem formulation such as (4.6), in which all policy variables are destined for model determined settings, we can condense the variable set S_I to the set of Mean Resupply Times at each location. All other operations control variables are then determined by the inventory model.

The ordering of the major stages in formulation (4.6) implies that the optimization over the product design variables occurs for all possible combinations of the operations control and manugistics design variables. In theory, this could include an extremely high number of possibilities, but, in reality, the number of combinations can be reduced. The set of feasible operations control variables can be reduced by assuming, as we explained above, that the decision maker is using a specific inventory policy to determine safety stock allocations and order quantities at each location once Mean Resupply Times are specified. Mean Resupply Times, then, suffice to summarize the decisions made in the outermost problem in (4.6) and consequently are used as state variables in the two inner optimizations. As the product configuration is built from the inner optimization, we can calculate the safety stock and order quantities for each operating and repair location based on product demand rate and the specified Mean Resupply Times.

This procedure of using Mean Resupply Time as a decision variable in the outer operations control problem and as a state variable for the inner product design problem, effectively breaks the dependency between Mean Resupply Time and demand. When a product configuration is chosen by the DP algorithm of the inner problem, the resulting distributions of demand for components and LRU's which correspond to the chosen configuration are then used to determine safety stocks and batch sizes that satisfy the vector of Mean Resupply Times that are set by the solution to the outer problem.

In the military manugistics system, there are only four levels of repair and a finite set of repair locations from which to choose when formulating a repair policy. Also, many maintenance facilities are configured to repair only certain 'families' of similar sub-assemblies due to unique equipment or labor skills that might be required to perform the repair action. Within a given manugistics system, there might only be one or two repair sites equipped to maintain a family of sub-assemblies. The same is also true for the manufacturing facilities. For any given component type, there might be two or three outside manufacturers who produce the component or there might be only two or three cells where the component can be produced internally.

Because of the small set of options for repair and manufacturing locations, each choice from the feasible set of these locations can be considered when optimizing over the product design choices. One repair alternative might be to repair sub-assemblies at the operating location and the depot, while another alternative might be to repair only at the manufacturer. The inner problem then uses this state information along with the capacities at each location to determine where a component or sub-assembly should be repaired or manufactured.

The choice of repair and manufacturing location directly impacts the inventory decisions of where and in what amount to stock component parts. For example, if repair occurs at the operating location, then we need to maintain enough sub-assemblies to replace failed ones taken off the product and we need to stock enough components to serve as replacements for the failed components that are revealed during repair. If repair occurs at a higher echelon, either the depot or manufacturer, then we only need to stock enough replacement sub-assemblies to cover the amount of time it takes to transport a failed assembly to the repair point and receive a serviceable one in return.

The innermost optimization over the product design variables in (4.6) can be separated into smaller stages which correspond to selecting the configuration option for each requirement of the product structure as shown in Figure 3.1. We propose to solve the innermost problem in (4.6) using a dynamic programming (DP) method that considers configuration requirements in stages. The configuration options are then built up, in turn, from their requirements. The decision at each of these stages involves choosing a specific configuration option to fill a requirement and become part of the product structure. Breaking down the product structure into sets of individual requirements allows us to 'build' the product design one requirement at a time. As with the calculation for MTBF shown in Chapter 3, the 'bottom-up' approach can also be applied when filling the configuration requirements with component and sub-assembly choices in the DP solution to the innermost optimization in (4.6). The DP solution procedure, then, can be viewed as a somewhat artificial 'building' of the product from the given options for configurations and components.

The return from each stage of the DP will be defined as the contribution to LCC and U, system unavailability. The cost and unavailability performance measures discussed in Chapter 3, will be calculated in stages, one configuration requirement at a time, using the information provided by the state variables at each stage. The state variables and their associated transformation functions are defined by analyzing the performance measures in greater detail. The overall objective function can be expressed by the equation, $LCC + \gamma U$. This function can be expanded as a combination of the performance measures as shown in equation (3.31).

Constituent Functions of the Objective Function

Availability

Since system level availability is derived from MTBF and MDT, the state variables must contain the information necessary to compute these measures. As configuration options are chosen, MTBF can be built starting from individual component MTBF's. MDT is a function of Mean Resupply Time, the feasible set of repair and manufacturing locations which are held as state variables from the outer problems, and Mean Corrective Maintenance Time (MCT) which is built up during the inner optimization over the product design decision variables.

Figure 4.3 shows the design structure, with all of the configuration options to be considered, for a generic product. The product structure, as we have represented it in Figure 4.3, is an arborescent network with the options as nodes. The top-level of the product structure corresponds to the single requirement A. Each requirement branches to a set of configuration

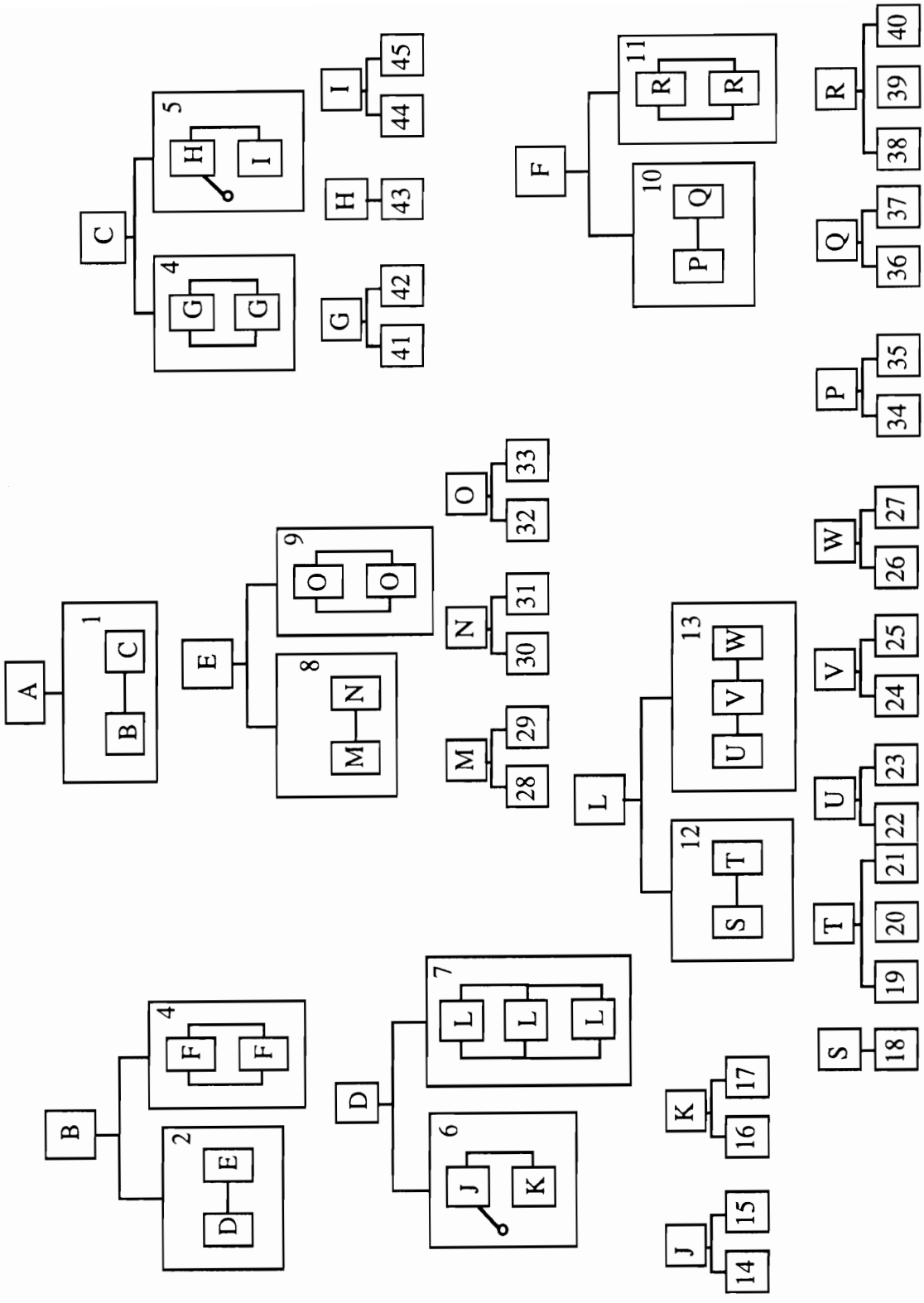


Figure 4.3: Product Structure Tree

options which represent decision points in designing the product. The decision maker must choose which option will be used to fill a specific requirement. We will begin at the bottom of the product structure to fill in the requirements and proceed upward along the branches. At any stage, the requirements which are 'below' the decision point have been determined by state-dependent decision rules while the requirements 'above' the decision point are yet to be determined. In Figure 4.3, if the decision point is at requirement L, then the requirements below it are S through W and the requirements above it are B through F.

As shown in Chapter 3, system level availability (A_S) can be calculated directly using LRU availabilities (A_l). We must look at the components which make up the LRU in order to compute LRU availability. We will use Figure 4.3 as an example to illustrate how an LRU's MTBF and MDT are built from its underlying component parts. Configuration 1 represents the top level of the product structure and consists of two requirements in series. Requirement B can be filled by either configuration option 2 or 3 and requirement C can be filled by either configuration option 4 or 5. Option 2 contains requirements in series, options 3 and 4 contain requirements in active redundancy and option 5 contains requirements in passive redundancy. The remaining options are built from one of the three configurations described above. We will designate requirements C, D, E, and F as LRUs. The lowest level of the product structure corresponds to the component level. Options 14 through 40 represent component types that either the manufacturer produces or can purchase from another contractor.

Using Figure 4.3 as an example, $MTBF_D = MTBF_6$ or $MTBF_7$, depending on which option in chosen in the product design. As examples, MTBF's for configuration options 6, 7 and 2 are computed from the formulas in Chapter 3 in the following manner;

$$MTBF_2 = \left(\frac{1}{MTBF_D} + \frac{1}{MTBF_E}\right)^{-1}$$

$$MTBF_6 = MTBF_J + MTBF_K$$

$$MTBF_7 = \sum_{i=1}^{3} \frac{1}{i} (MTBF_L) = \left(\frac{11}{6}\right) (MTBF_L), \text{ where}$$

$$MTBF_L = \left(\frac{1}{MTBF_S} + \frac{1}{MTBF_T}\right)^{-1} \text{ or } \left(\frac{1}{MTBF_U} + \frac{1}{MTBF_V} + \frac{1}{MTBF_W}\right)^{-1}$$

finally,

$$MTBF_J = \frac{1}{\lambda_{14}} \text{ or } \frac{1}{\lambda_{15}}$$

$$MTBF_K = \frac{1}{\lambda_{16}} \text{ or } \frac{1}{\lambda_{17}}$$

and the other component requirements are similarly specified, then

$$A_{2n} = \frac{MTBF_2}{MTBF_2 + MDT_{2n}}$$

Cost Elements

Even though we defined many specific variables and performance measures for *LCC* in Chapter 3, for the development of the DP formulation that follows, it is convenient to represent the *LCC* function in the following general format;

$$\sum_{ij} f_{ij} (S_1; z_k) \delta_j(S_2) \delta_i(S_3) \tag{4.7}$$

where i is a subscript identifying a component type, j is a subscript identifying a facility, z_k is the state vector at stage k, S_3 is the set of product design variables and S_2 is the set of manugistics

design variables and S_1 is the set of operations control variables, respectively. The term facility refers to repair facilities as well as the manufacturing cells.

$$\delta_i(S_3) = \begin{cases} 0 \text{ if component } i \text{ is not used} \\ 1 \text{ if component } i \text{ is used} \end{cases}$$

$$\delta_j(S_2) = \begin{cases} 0 \text{ if facility } j \text{ is not used} \\ 1 \text{ if facility } j \text{ is used} \end{cases}$$

The presence of $\delta_i(S_3)$ and $\delta_j(S_2)$ in (4.7) indicate that component i is included in the product design and that facility j is included in the manugistics design, and there is an additive contribution to the *LCC* function, otherwise no cost is incurred. The amount of the cost that is introduced for a stage k is;

$$LCC_k = \text{production cost} + \text{acquisition cost} + \text{transportation cost} + \text{inventory cost} + \text{disposal cost} + \text{repair cost} \quad (4.8)$$

In every case, the cost functions are additive over the requirements in the product structure. Each component type has a set of associated costs that are added to the LCC when it is chosen to be included in the product design. With the exception of Transportation Cost, all of the cost equations described in Chapter 3, are functions of the design choices for the sets of requirements. Because each component type will be manufactured, repaired, inspected and stocked separately, the LCC is additive over the individual types.

The interaction of the product design variables in the cost functions occurs through the sharing of capacity. As configuration options are included in the product design, they are assigned to the various support facilities which increases the workload at those sites. Because support facility

capacities may dictate which repair or manufacturing facility will be used to support a specific configuration option, it also determines the associated costs that are included in the *LCC* function.

Stages and State Variables

The requirements designated as LRU's divide the product structure into two segments. A decision for requirements above the LRU level in the product structure only involves choosing an option to fill the requirement while decisions below the LRU level also include, as state variables, the specification of the feasible set of repair and production locations for the components contained within the option. As we proceed upward through the branches of the product structure, we will eventually reach the requirements designated as LRUs. At this point, the decision concerning repair and production have been resolved and the information in the state variables, for these decisions, is no longer required. The requirements which are LRUs serve as a transition point within the DP framework with regard to information needed as state variables. This translates into a two-phase process within the DP for building A_S, with the LRUs as the transition point between the phases.

The stages of the DP formulation of the product design sub-problem are the decision points in the artificial building of the product structure described above, at which the solution method must choose between alternate configuration options or component types. The manner in which the stages of the DP formulation of this design problem are defined is a natural result of the arborescent nature of the product structure network. For the design problem depicted in Figure 4.3, stage 1 would correspond to choosing between component types 38, 39 and 40 to fill

requirement R and the last stage would correspond to choosing between options 4 and 5 to fill requirement C. Table 4.1 shows all of the stages for Figure 4.3. Note that the sequence of stages is not unique. At the end of this section, we explain the conventions that were used to determine the sequence of stages shown in Table 4.1.

For now, it is important to note that the DP algorithm treats product structure requirements above the LRU's (Phase 2) somewhat differently from those below the LRU level (Phase 1). In Phase 1, as we consider using a particular component type to fulfill a specific functional requirement at stage k, we will define the state vector of the product design sub-problem to be, $\mathbf{z}_k$ = (**a**, **b**, **c**, **d**, **e**, **f**, **g**) where **a** is defined as the vector of configuration requirement failure rates, **b** is defined as a vector of configuration requirement MCT's, **c** is defined as a vector of configuration requirement demand rates, **d** is defined as a vector of repair and manufacturing facility capacities, **e** is a set of repair locations provided by the middle optimization in (4.6), **f** is a vector of Mean Resupply Times provided by the outer optimization in (4.6) and **g** is a vector of configuration requirement *LCC's*. In Phase 2, system availability is calculated from the LRU availabilities instead of from failure rates and Mean Down Times. Consequently, for Phase 2, we define α_* to be the vector of system availability at each location. We will describe these state variables in greater detail later in this chapter.

The decision to choose a specific configuration option to enter the product structure is based on the contribution that option makes to A_S and *LCC*. Options are chosen to fill requirements which may be contained in other options throughout the product structure to the top level. The contribution that an individual configuration option makes to A_S depends on the configuration of of the requirements at the higher levels in the product structure from which it has branched. The

Table 4.1

Sequence of Stages

Sequence	Stage	Options	h	New State	Current State
1	H	43/44,43/45	1	a_H	a_C
2	G	41,42	2/3	a_G	a_C
3	C	4,5	0	a_C	a_R or a_P or a_U or a_S or a_T
4	R	38,39,40	4/9	a_R	a_F
5	Q	36,37	2/3	a_Q	a_F
6	P	34,35	2/3	a_P	a_Q
7	F	10,11	0	a_F	a_B
8	O	32,33	2/3	a_O	a_E
9	N	30,31	1	a_N	a_E
10	M	28,29	1	a_M	a_N
11	E	8,9	0	a_E	a_B
12	W	26,27	6/11	a_W	a_L
13	V	24,25	6/11	a_V	a_W
14	U	22,23	6/11	a_U	a_V
15	T	19,20,21	6/11	a_T	a_L
16	S	18	6/11	a_S	a_T
17	L	12,13	0	a_L	a_D
18	J	14/16,14/17 15/16,15/17	1	a_J	a_D
19	D	6,7	0	a_D	a_E
20	B	2,3	0	a_B	a_A
21	A	1	0	a_A	

following formulas show how the elements of the state vector are used to calculate the contribution of the option under consideration to the system performance;

$$\frac{1}{MTBF_s} = a_r + h\frac{1}{MTBF_o} \tag{4.9}$$

where the subscript S indicates the system MTBF and o is the current option under consideration. The subscript r represents the requirement from which option o has branched. The element a_r summarizes the value of the failure rate for the system at the point when option o is being considered to be included in the product design.

Every component that is included in the product design contributes to the overall failure rate of the product in some fashion. A component can either increase or decrease the failure rate of the product depending on whether it is added to the product in series with the other components, in parallel with them or as a standby unit. This contribution is reflected in the value of h in equation (4.9). As we proceed down any branch of the product structure network, we can determine the exact configuration in which each component at a terminal node is included in the design and therefore, we can determine the component's contribution to failure rate.

Every terminal node has a unique value of h that can be calculated prior to starting the DP. This value of the constant h depends on the configuration of the requirements from which option o has branched. As discussed earlier, a module with N elements in active redundancy inflates the MTBF of the module by a factor of $(\sum_{i=1}^{N}\frac{1}{i})$. We must then decrease the failure rate by this same amount or, $(\sum_{i=1}^{N}\frac{1}{i})^{-1}$. The failure rate of each terminal node must be decreased by the

amount $(\sum_{i=1}^{N}\frac{1}{i})^{-1}$ for every node above it which is in a parallel configuration. Therefore, for every terminal node, h is multiplied by $(\sum_{i=1}^{N}\frac{1}{i})^{-1}$ for every option above it with N modules in active redundancy. The value of h is set equal to 1 for any option which is a terminal node and does not have any options in active redundancy in its path to configuration 1. Finally, the constant h is set equal to 0 for every configuration option which is not a terminal node for the product structure.

For example, options 4 and 11 are not terminal nodes and so have values of 0 for h. Option 38 is a terminal node with requirement R in parallel, and requirement F in parallel on its path to configuration 1. If component type 38 is the option under consideration, then h is multiplied 2/3 for requirement F and 2/3 for requirement R and so will equal 4/9 as shown in Table 4.1.

Now we examine Mean Corrective Time as a state variable.

$$MCT_{\ell R} = \frac{(\lambda_o)(w_o)(MCT_o)+b_r}{(\lambda_o)(w_o) + c_r} \tag{4.10}$$

where r is a requirement within the configuration for LRU ℓ and o is an option for this requirement. Once again, at a given decision point, we wish to assess the contribution that each option will make to the MDT of the system. Similar to a_r, b_r summarizes the weighted value of MCT for the system at the point when option o, within LRU ℓ, is being considered and c_r summarizes all the values of the weights. This calculation follows from the equation (3.8) of Chapter 3.

We define $d_{n,r}$ as the load at location n at stage r in the DP.

$$RL_n = d_{n,r} + D_{oln}\mu l_{ol} \tag{4.11}$$

Equation (4.11) shows how the load of repair location n is increased by the repair load of configuration option o when it is assigned to location n for maintenance.

$$PL_p = d_{p,r} + D_o[P_{op} + \frac{S_{op}}{q_o}] \tag{4.12}$$

Equation (4.12) shows how the load for manufacturing cell p is increased by the production load of configuration o when it is assigned to cell p for production. This equation follows from equation (3.7) of Chapter 3. The loads for each location are tracked by the model to ensure that capacity at any facility will not be exceeded.

The range of values for Mean Resupply Time, by echelon, to be considered by the outer optimization in (4.6) will be supplied by the decision maker. For the innermost optimization problem in (4.6), these Mean Resupply Times, represented by the vector **f**, will be held constant. The batch sizes and safety stocks for each option under consideration at stage r, will be adjusted to maintain this fixed Resupply time. The values for the vectors **e** and **d** are also fixed for the innermost optimization problem. These values will nevertheless influence the options that are chosen by the innermost problem as well as the performance measures of *LCC* and Availability.

$$LCC_S = LCC_o + g_r \tag{4.13}$$

Equation (4.13) shows the additive nature of the *LCC* for the system, where the subscripts are defined as above. As in equation (4.9), g_r is the value of the *LCC* for the system at the point when option o is being considered in the product design.

In Phase 1 of the DP algorithm, the vectors **a**, **b**, **c**, **d**, **e** and **f** are all necessary to determine LRU availability ($A\ell$) as described by the hierarchy of performance measures in Figure 3.4. The component information is maintained by these state variables during Phase 1. Vector **a** contains the MTBF information and vectors **b** through **f** are necessary to compute MDT. These two measures are then combined to calculate availability of the LRU. Once we reach the LRU level in the product structure, it is no longer necessary to use the component information. For this reason, we use the vector α to be LRU availabilities at each location and we use this vector, along with **g**, for LCC, to build the decision rules. The decision rules will be discussed in greater detail in a later section.

Although the dimensionality of the state vector, $\mathbf{z_k}$, is high, each individual stage requires only a small subset of the vector elements to determine values for the decision variables at that stage. When computing system level MTFB for example, only those elements of the vector **a** that correspond to the configuration option under consideration are relevant to the decision variable calculations. Table 4.1 shows the complete list of state variables, values for the constant h and configuration options that are associated with each stage for the product structure in Figure 4.3.

Because of the nature of the product structure in Figure 4.3, certain elements of the state vector will equal each other. For example, the element a_A is carried through to the next stage and is equal to a_B because, as shown in Table 4.1, the value of h is zero at that point. Transforming the state information at decision points where h is equal to zero, then becomes a trivial matter but it is done to preserve the flow of state variable information from requirement to requirement.

Due to the arborescent nature of the product structure, there is no unique sequence for the decision points or stages of the DP. Also, paths which branch from the same decision point are what we shall call 'linked' at that point and can be processed in parallel. For example, options 2 and 3 both branch from requirement B and so they are linked at requirement B. Any decision points which are linked require the same set of state variable information to be made available from the parent node. A set of rules must be established to ensure the correct information is available at each decision point.

We will number the requirements of options that are in a series configuration from left to right to establish the sequence of building the links for any branches which must be processed in parallel in the product structure. Using Figure 4.3 as an example, requirement D in configuration option 2 would be counted as the first requirement in the configuration while requirement E would be counted as the second one. For option 1, requirement B is the first and C is the second.

Since configuration 1 is always a set of requirements in series, we can begin the DP recursion along the branches of either the first (farthest to the left in Figure 4.3) or last (farthest to the right) requirement in configuration 1. We will arbitrarily choose the last requirement and proceed down its branches to the terminal nodes to begin building the decision rules.

We will use the following set of rules to progress from one stage to another within the product structure. As mentioned above, we will begin with the last requirement in configuration 1 and proceed down its branches to the terminal nodes. If we arbitrarily sequence the children of a particular requirement from left to right, we will proceed down the rightmost path until we reach

a terminal node. This will be the first stage to begin building the decision rules and in this example, it would be stage H. We then move to the next sibling on the left and proceed along its branches until we reach its terminal nodes. The parent requirement of these terminal nodes is the next stage. In Figure 4.3, option 4 is the next sibling to the left of option 5, and its terminal nodes are options 41 and 42. Requirement G is the parent of these options and is the next stage in the sequence. Once all of the siblings have been visited, we move upward to their parent node for the next stage. In Figure 4.3, options 4 and 5 are the only siblings with requirement C as the parent, which then becomes the next stage. We continue traveling upward, until we the parent requirement is contained in configuration 1. We then move to the requirement on the left (B) and continue in the same fashion. Following the branches of the rightmost option under requirement B, we reach requirement R as the next stage. Table 4.1 lists the entire sequence of stages corresponding to Figure 4.3.

For requirements in a series configuration, there is an additional link between the requirements which must be maintained. If we continue to use the left to right numbering scheme, then all of the branches for requirement *n* must be travelled before branching to requirement *n-1*. There is a link between the final stage of each path which branches from requirement *n-1* and the requirement *n* in the series configuration. For example, in Figure 4.3, if requirement B is the first requirement and C is the second, there is a link between the decision point for requirement D and the decision point for C as well as the decision point for F and for C.

Transformation Functions

The following set of equations describe how the state information is transformed from one requirement to the next in the DP formulation. The information carried in the vector **a** at a given requirement is used to calculate the MTBF for the LRU after an option is chosen for the requirement. The information provided by the vectors **b**, **c**, **d**, **e** and **f** are used to determine the MDT for the system. The vector **g** is used to generate *LCC* for the product and α_n is used for LRU availability by location.

First we consider Phase 1 of the DP algorithm. The transformation function for MTBF is based on its reciprocal, failure rate. We must then determine the failure rate for the option under consideration, to determine its contribution to total system failure rate. For the series configuration, the failure rates are additive and can be directly added to the total failure rate. For the active redundant configuration, we multiply the MTBF of the option under consideration by appropriate amount and add it's reciprocal to the total failure rate. The passive redundant configuration is a special case.

When a system of modules is in a passive redundant configuration and it is assumed that the modules in stand-by do not fail when they are not operating, the expected total system life is the sum of expected module's lives. To compute the failure rate for the passive redundant configuration, we must first add the MTBFs of its modules and then take the reciprocal. To simplify the transformation function for this case only, we will consider the entire passive redundant configuration as one stage and fill the configuration requirements simultaneously.

The transformation function that is used to update the system MTBF with each configuration choice is;

$$a_{r'} = a_r + h\frac{1}{MTBF_o} \tag{4.14}$$

where o is the option under consideration for requirement r'. The subscript r represents one of the following cases which are illustrated below using Figure 4.3. Case I identifies the situation in which requirement r is the parent requirement from which option o has branched. Case II identifies the situation in which requirement r is the requirement to the right of requirement r' of a series configuration assembly. Case III identifies the situation in which requirement r is a requirement which branches to a terminal node on a path which originated from a requirement in a series configuration. This last case occurs because the terminal nodes for a requirement r which is in a series configuration, are linked to the next requirement which is to the right of requirement r. The following equations represent each of the above cases;

Case 1: $a_D(6) = a_B + h\frac{1}{MTBF_6}$

Case 2: $a_E(8) = a_D + h\frac{1}{MTBF_8}$

Case 3: $a_C(4) = a_T + h\frac{1}{MTBF_4}$

Since MCT is a weighted average, we need to accumulate the component MCTs and the values of the weights in order to update the state variables b_r and c_r. The corresponding transformation functions are;

$$b_{r'} = b_r + (\lambda_o)(w_o)(MCT_o) \tag{4.15}$$

$$c_{r'} = c_r + (\lambda_o)(w_o) \tag{4.16}$$

where r is the stage of the DP and o is the next configuration option under consideration. Once we reach the requirements in the product structure that have been designated as LRUs, we enter what we shall call Phase 2 of the DP recursion. At this point, we no longer use **a** from equation (4.14) and we shift to using α_n in the state vector . Since the repair location assignment and production control policies have been resolved at earlier stages (i.e. the outer problems in (4.6)), elements **d**, **e** and **f** of the state vector are no longer required to make configuration choices at points in the product structure above the LRU level. The transformation then becomes;

$$\alpha_{r'n} = (\alpha_{r\,n})\ (\alpha_{t\,n}) \tag{4.17}$$

The transformation functions for equations (4.11) and (4.12) are as follows;

$$d_{n,r'} = d_{n,r} - D_{o\,t\,n}\ \mu 1_{ot} \tag{4.18}$$

for repair facilities, and

$$d_{p,r'} = d_{p,r} + D_o[P_{op} + \frac{S_{op}}{q_o}] \tag{4.19}$$

for production facilities.

The transformation function for equation (4.13) is;

$$g_{r'} = g_r + LCC_o \tag{4.20}$$

where LCC_o is computed using equation (4.9).

Decision Rules

The DP solution is based on a set of decision rules, conditioned on the state of the system, that determine choices from among the alternate options at any decision point in the problem. Each decision rule for the inner product design problem of (4.6) chooses a configuration option which

will minimize the objective function based on the information contained in the elements of the state vector, $\mathbf{z}_k$. Using a backwards recursion, we build decision rules for each decision point starting from the bottom of the product structure. Once the decision rules have been determined, we fill in the requirements beginning at the top level and continue along the branches of the product structure until we choose options for all of the requirements.

The objective function consists of two performance measures, life cycle cost and system unavailability, as well as a parameter, γ, which is assigned a value by the decision maker. In mathematical terms, the objective function is $LCC + \gamma U$, which we want to minimize. When an option is included in the product design, its *LCC* is added to the system *LCC* and its contribution to *MTBF* and *MDT* is incorporated into the current *MTBF* and *MDT* of the system.

For any requirement r, the *LCC* and availability of the system can be viewed as consisting of the *LCC* and availability of the requirements above r, the *LCC* and availability of requirement r and the *LCC* and availability of the requirements below r. The *LCC* and availability above requirement r is contained in elements α_n and $\mathbf{g}$ of the state vector $\mathbf{z}$ while *LCC* and availability below r is represented by LCC^*_{k-1} and A^*_{k-1}. At each requirement, we wish to find the optimal value of o, which we shall call o^* which minimizes the objective function for all requirements. Let $\mathbf{g}_k$ represent the *LCC* resulting from choices made for stages K through k, or $\mathbf{g}_k = \sum_{i=k+1}^{K} LCC_i$ and let $\alpha_{nk} = \prod_{i=k+1}^{K} A_{in}$ represent the availability of these choices, then we must consider the following objective function in order to build the decision rules for the requirement at stage k;

$$g_k + LCC_r(c_k,o) + LCC^*_{k-1}(z_{k-1},o) + \gamma[1 - \frac{\sum_{n \in J8} N2_n \ \alpha_{n\,k} \ A_{r\,n}(o,f) A^*_{k-1,n}(z_{k-1},o)}{\sum_{n \in J8} N2_n}]$$

where r is the requirement at stage k and o is the option under consideration. As can be seen by the above equation, the optimal choices for stages 1 through k-1, depend only on the stage information at stage k-1 which was transformed from stage k by choosing option o. The inventory portion of LCC of the requirement must consider the increase in demand rate of including option o if it represents a component which is used elsewhere in the product, hence the dependence of LCC_r on $\mathbf{c_k}$. $A_m(o,f)$ is defined as the contribution to system availability at locaiton n of option o. As with all availability figures, A_m is a function of mean LRU resupply times, hence the dependence of A_m of on $\mathbf{f}$. The availability portion of the objective function is the fleetwide system availability across all operating locations. The expression above for the objective function illustrates that the principle of optimality holds for this DP formulation.

The transformation function is of the form;

$$z_{k-1} = T(z_k, o) \tag{4.21}$$

and the objective function can be stated as;

$$g_k + LCC_r(c_k,o) + LCC^*_{k-1}(z_k,o) + \gamma[1 - \frac{\sum_{n \in J8} N2_n \ \alpha_{n\,k} \ A_{r\,n}(o,f) A^*_{k-1,n}(z_k,o)}{\sum_{n \in J8} N2_n}]$$

Since the values of $\mathbf{d}$, $\mathbf{e}$, $\mathbf{f}$ for the state vector $\mathbf{z_k}$ are supplied by the outer and middle optimization problems of (4.6), they are constant for each stage of the inner product design problem.

When deciding between two options, 1 and 2, to fill a requirement, the objective function is transformed based on the contribution of the option under consideration. Let option 1 transform the objective function to;

$$g_k + LCC_r(c_k,1) + LCC^*_{k-1}(z_k,1) + \gamma[1 - \frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ A_{r\,n}(1,f)A^*_{k-1,n}(z_k,1)}{\sum_{n\in J8} N2_n}]$$

and option 2 transform the objective function as follows;

$$g_k + LCC_r(c_k,2) + LCC^*_{k-1}(z_k,2) + \gamma[1 - \frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ A_{r\,n}(2,f)A^*_{k-1,n}(z_k,2)}{\sum_{n\in J8} N2_n}]$$

The point of indifference between option 1 and option 2 occurs when;

$$LCC_r(c_k,1) + LCC^*_{k-1}(z_k,1) - [LCC_r(c_k,2) + LCC^*_{k-1}(z_k,2)]$$
$$+ \gamma[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f)A^*_{k-1,n}(z_k,1) - A_{r\,n}(2,f)A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}] = 0$$

or when

$$LCC_r(c_k,1) + LCC^*_{k-1}(z_k,1) - [LCC_r(c_k,2) + LCC^*_{k-1}(z_k,2)] =$$
$$\gamma[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f)A^*_{k-1,n}(z_k,1) - A_{r\,n}(2,f)A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}]$$

Option 1 will minimize the objective function when;

$$LCC_r(c_k,1) + LCC^*_{k-1}(z_k,1) - [LCC_r(c_k,2) + LCC^*_{k-1}(z_k,2)] <$$
$$\gamma[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f)A^*_{k-1,n}(z_k,1) - A_{r\,n}(2,f)A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}] \qquad (4.22)$$

and similarly, option 2 will minimize the objective function when;

$$LCC_r(c_k,1) + LCC^*_{k-1}(z_k,1) - [LCC_r(c_k,2) + LCC^*_{k-1}(z_k,2)] > \gamma\left[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f)A^*_{k-1,n}(z_k,1) - A_{r\,n}(2,f)A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}\right] \quad (4.23)$$

Thus, when two options are being considered to fill a requirement, if the inequality in equation (4.22) occurs, we choose configuration option 1. If the inequality is reversed, as in equation (4.23) we choose configuration option 2.

Each decision point, or requirement, in the product structure can be categorized into one of three different cases. The decision rule for the first two cases will be a special form of the generalized rules described in equations (4.22) and (4.23). The decision rule for case three is of the general form.

Case I: Requirement r branches only to a terminal node

In this case, there are no requirements below requirement r or in series with requirement r, so $LCC^*_{k-1}(z_k,o)$ takes on the value of zero and $A^*_{k-1,n}(z_k,o)$ takes on the value of 1. The decision rule then reduces to;

option 1 minimizes the objective function when;

$$LCC_r(c_k,1) - LCC_r(c_k,2) < \gamma\left[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f) - A_{r\,n}(2,f)]}{\sum_{n\in J8} N2_n}\right] \quad (4.22a)$$

option 2 minimizes the objective function when;

$$LCC_r(c_k,1) - LCC_r(c_k,2) > \gamma\left[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A_{r\,n}(1,f) - A_{r\,n}(2,f)]}{\sum_{n\in J8} N2_n}\right] \quad (4.23a)$$

Case II: Requirement r is a linking requirement

In this case, requirement r serves as a 'dummy' stage which merely carries the state information from stage $k+1$ through stage k intact. For this reason, $LCC_r(c_k,o)$ takes on the value of zero and $A_r(o,f)$ takes on the value of 1. The decision rule then reduces to;

option 1 minimizes the objective function when;

$$LCC^*_{k-1}(z_k,1) - LCC^*_{k-1}(z_k,2) < \gamma\left[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A^*_{k-1,n}(z_k,1)-A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}\right] \qquad (4.22b)$$

and option 2 minimizes the objective function when;

$$LCC^*_{k-1}(z_k,1) - LCC^*_{k-1}(z_k,2) > \gamma\left[\frac{\sum_{n\in J8} N2_n\ \alpha_{n\,k}\ [A^*_{k-1,n}(z_k,1)-A^*_{k-1,n}(z_k,2)]}{\sum_{n\in J8} N2_n}\right] \qquad (4.23b)$$

Case III: Requirement r is in series with stage $k+1$

For this case, the decision rule does not reduce any further and is of the form described in equations (4.22) and (4.23).

The recursion for the objective function is not strictly additive or strictly multiplicative. If we combine the two measures of LCC and availability and define the resulting function to be;

$$f_k = LCC^*_k + \gamma\left[1 - \frac{\sum_{n\in J8} N2_n\ A_{kn}}{\sum_{n\in J8} N2_n}\right]$$

for stage k, then the recursion becomes;

$$f_k = min_{o \in J1,J2,J3} \{ g_k + LCC^*_{k-1}(z_{k-1}) + LCC_r(c_k,o) + \gamma[1 - \frac{\sum_{n \in J8} N2_n \ \alpha_{n\,k} \ A_{r\,n}(1,f) A^*_{k-1,n}(z_k,1)}{\sum_{n \in J8} N2_n}] \}$$

where $LCC_r(c_k,o)$ is the *LCC* of option o at requirement r and $A_r(o,f)$ is the availability of option o at requirement r. LCC^*_{k-1} and A^*_{k-1} are defined as above.

Incorporated in $LCC^*_{k-1}(z_{k-1})$ and $A^*_{k-1}(z_{k-1})$ are the decision rules which were constructed at stage *k-1, k-2...1*. Each of these decision rules is based on the state information which was provided to a stage by the next stage. For example, each alternate choice for stage k will transform the state information from $\mathbf{z_k}$ to $\mathbf{z_{k-1}}$, using equation (4.21), which is then provided to stage *k-1*. The decision rule for stage *k-1* then uses the transformed state information to make the optimal choice for stage *k-1* and the process is continued in this fashion.

Example Problem

The following example is presented to illustrate how the DP framework that was described in the preceeding section can be used to solve the optimization problem in (4.6). Figure 4.4 shows an example of a product structure and the choices that are available for each configuration requirement. Requirements B and C will be defined as the LRU's for Figure 4.4 Table 4.2 shows a list of the stages that result from the product structure in Figure 4.4, their associated state variables and computed values of h to be used with equation (4.21). For simplicity, only one operating location will be used in this example.

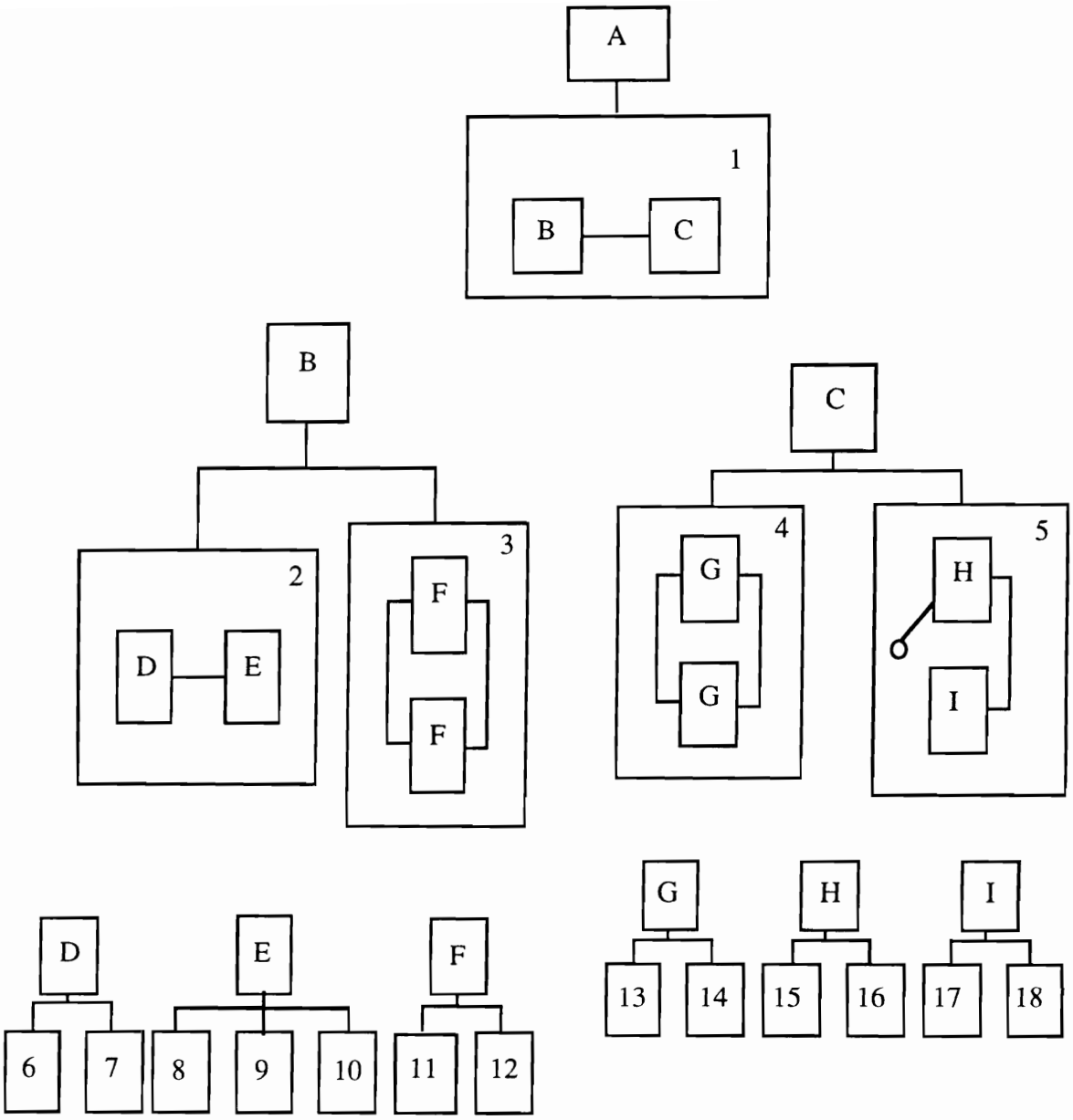


Figure 4.4
Example Product Structure

Table 4.2

Sequence of Stages

Sequence	Stage	Options	h	New State	Current State
1	H	15/17,15/18 16/17,16/18	1	a_H	a_C
2	G	13,14	2/3	a_G	a_C
3	C	4,5	0	a_C	a_F or a_D
4	F	11,12	2/3	a_F	a_B
5	E	8,9,10	1	a_E	a_B
6	D	6,7	1	a_D	a_E
7	B	2,3	0	a_B	a_A
8	A	1	0	a_A	

Configuration 1, in Figure 4.4, represents the top level of the product structure. The goal is to determine which underlying configuration options should be included in the product to minimize *LCC* and *U*, system unavailability. Working upward from the bottom level of the product structure in the backward recursion of the DP formulation, we determine the decision rules for each configuration option that minimizes the objective function, $LCC + \gamma U$, for each system state. This procedure is continued until decision rules for all of the requirements are set. Once we reach the final stage, the state of the system is known and we then proceed in the opposite direction and decide which configuration options will fill the requirements.

Using component type 6, which is repaired at operating location *n* and produced in manufacturing location *m* as an example, the following set of calculations are performed prior to the DP in order to construct a parameter database for the DP;

1. $MTBF_6 = 1/\lambda_6$
2. L_6 = production time$_6$ + setup time$_6$ + inspection time$_6$
3. $ML_6 = L_6 + \omega_m$

   where ω_m is the average wait time for production facility *m*
4. LCC_6 = sum of all costs

These same calculations would be performed for the other component types in the product structure to begin building system level MTBF and MDT. Table 4.3 summarizes all of these computations for each component type in Figure 4.4 and Table 4.4 contains the individual pieces that make up the total *LCC* for each component type.

Table 4.3

Component Type Performance Measure

Comp Type	Stage	MDT	MTBF	Avail	MCT	λ_i	ST	RL	PL
6	D	37.0	1100	0.967	105	0.000909	378	120	170
7	D	79.8	450	0.849	130	0.002222	174	177	46
8	E	89.4	950	0.914	145	0.001053	198	197	78
9	E	103.4	1500	0.936	215	0.000667	400	267	211
10	E	163.6	850	0.838	220	0.001176	268	332	103
11	F	44.4	705	0.941	70	0.001418	106	97	32
12	F	67.2	900	0.931	34	0.001111	182	111	69
13	G	22.4	578	0.963	280	0.001731	329	312	106
14	G	12.4	975	0.987	320	0.001025	460	337	197
15/17	H	41.4	1590	0.975	85	0.000629	136	107	65
15/18	H	59.4	1240	0.954	140	0.000806	122	177	50
16/17	H	102.8	1650	0.941	95	0.000606	139	172	68
16/18	H	107.2	1300	0.924	100	0.000769	111	187	46

Table 4.4

LCC Component Functions

Comp Type	Prod Cost	Repair Cost	Quality Cost	Disp Cost	Trans Cost	Inven Cost	Total Cost
6	3700000	540000	110000	550	5700	120	4356370
7	525000	1625000	15750	4500	13750	50	2184050
8	975000	875000	115000	550	6500	75	1972125
9	1250000	810000	310000	480	4200	135	2374815
10	640000	1450000	850000	1500	7500	120	2184120
11	630000	575000	25000	2500	9000	60	1241560
12	750000	215000	40000	400	5500	50	1010950
13	600000	2500000	25000	1500	13000	30	3548625
14	1600000	1850000	90000	1100	7500	25	3548625
15/17	1750000	1350000	210000	500	4800	45	3315345
15/18	1335000	1600000	30000	1000	6500	30	2972530
16/17	1365000	1360000	50000	1200	4500	50	2780750
16/18	1157000	1470000	25000	1500	5000	40	2658540

As was previously discussed, when an option, such as choice 5 for requirement C, consists of a passive redundant configuration, all of the requirements in the option will be taken together as one stage. For this case only, each combination of underlying configuration options will be enumerated and their associated information combined to facilitate the build up of the MTBF portion of Availability. Using Figure 4.4 for this example, the combinations of 15 - 17, 15 - 18, 16 - 17 and 16 - 18 would all be examined to fill requirements H and I. Their combined information is represented as component types 15/17, 15/18, 16/17 and 16/18 in Table 4.3. Stages which correspond to requirements in a stand-by configuration will be referred to by the name of the primary requirement. For the remainder of the example, the stage for requirements H and I will be called Stage H.

Prior to starting the DP recursion, certain configuration options can be eliminated by comparing the individual component's *LCC* and availability. If a component has a higher availability and a lower cost than a second component, the second one is dominated and should be eliminated as an alternative. Before beginning the DP recursion, certain options from Table 4.3 can be eliminated. For example, in Table 4.3, option 8 dominates option 10 and so the set of alternatives for requirement E would then be restructured to eliminate option 10.

The top level of the product structure which corresponds to configuration 1, will always be represented by a set of requirements in series. Each requirement in configuration 1 is a parent to a set of options that will branch to other options or branch to component types. As mentioned earlier, each requirement is a decision point and will have a stage in the DP which correcsponds to it. We begin at the terminal nodes, which represent the component types, and proceed upward through the branches of the product structure to build *decision rules* for each stage. When we

reach the last stage in the DP, we reverse directions and using the intial values of the state variables, begin making *decisions* at each stage. Using Figure 4.4 and Tables 4.2 through 4.5 as references, the following sequence of events will occur when building the performance measures of system level Availability and *LCC*.

Certain elements of the vectors **a** through **g** in the state space $\mathbf{z_k}$ must be loaded with initial values before we begin the recursion for the innermost optimization problem of (4.6). There is an element in the vectors **a**, **b**, **c**, and **g** for every requirement/option combination in the product structure. The elements a_A, b_A, c_A, g_A, will all be initialized to values of zero since requirement A is the first requirement in the product and therefore, the last stage for the DP. It will then serve as the starting point when we move downward throught the product structure to begin making decisions. The element A_S will be initialized to a value of 1 for the decision rules.

The initial values for the elements of **d** are determined by the input parameters of capacity and load for each of the production and repair facilities in the manugistics system. The initial value for $d_{p,A}$ is set to PL_p for the production facility p and for $d_{n,A}$ is set to $L2_n$ for repair facility n. Table 4.6 shows the capacity and load for the 5 production facilities and 2 repair facilities which are included in this example problem. Elements **e** and **f** of the state vector are determined by the outer and middle optimization problems in (4.6). From Table 4.5, repair facility 6 is associated with LRU B and repair facility 7 is associated with LRU C and are the values of the vector **e** that are used in this problem. We will start with production facility 1 for LRU B and production facility 4 for LRU C. The initial value of each component of **f** is set to 0.8.

Table 4.5
Production and Repair Facilities

Comp Type	Production Facility	Repair Facility
6	1,2,3	6
7	1,2,3	6
8	1,2,3	6
9	1,2,3	6
10	1,2,3	6
11	1,2,3	6
12	1,2,3	6
13	4,5	7
14	4,5	7
15	4,5	7
16	4,5	7
17	4,5	7
18	4,5	7

Table 4.6
Facilities and Capacities

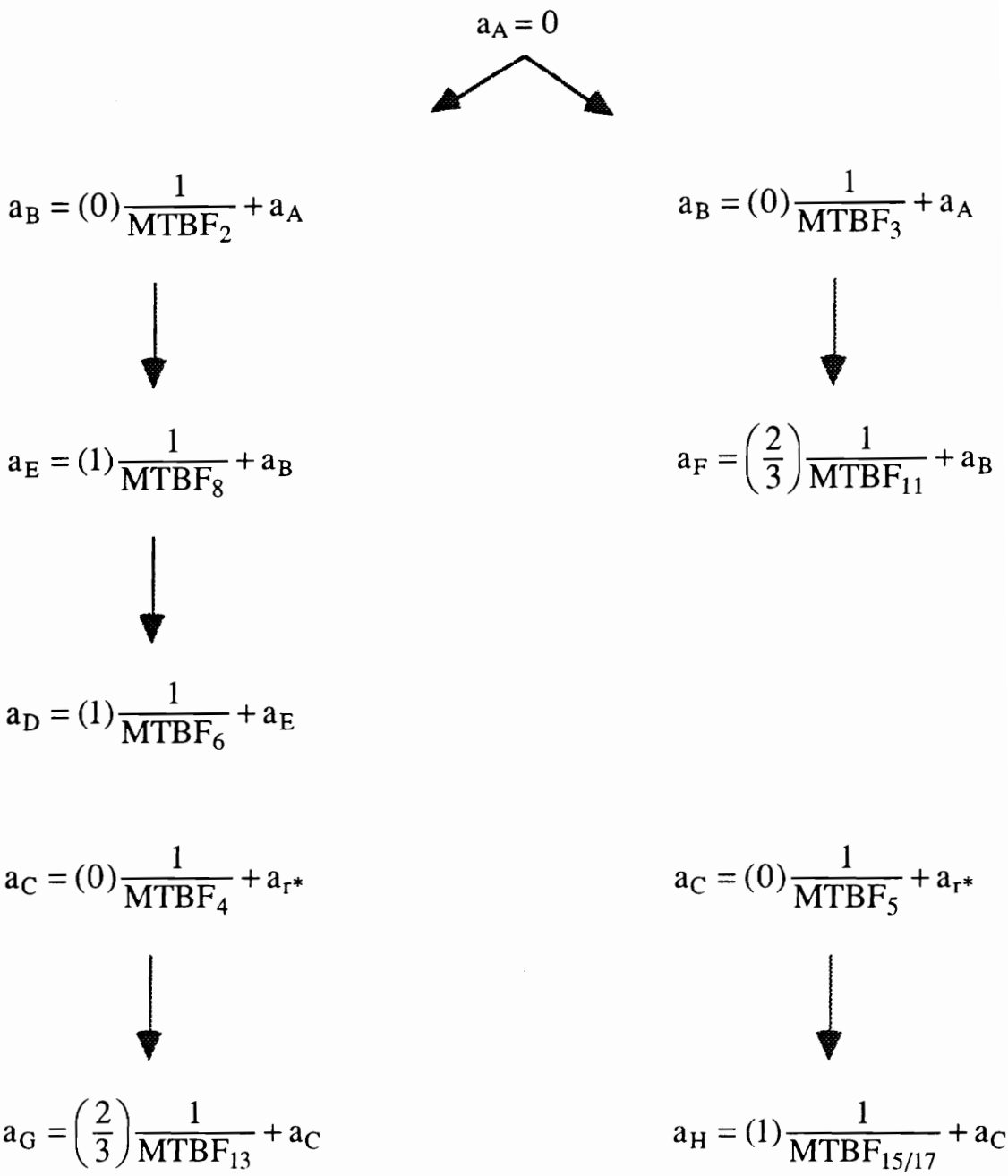
Facility	Total Capacity	Current Load	Available Capacity
1	2000	1200	800
2	3000	1800	1200
3	2500	1000	1500
4	3400	1700	1700
5	1500	1000	500
6	2500	1000	1500
7	3200	1200	2000

Prior to starting the DP recursion, a parameter database for the model is constructed which includes the values of h for equation (4.21), the sequencing of the stages, the value of γ, and the product structure links for the transformation functions. Tables 4.2 - 4.6 and Figure 4.5 represent the information which resides in this database. The cost information and the value of γ has been scaled down by a factor of 100,000 to keep the objective function values from becoming too large. For the remainder of the example, we will use the scaled value of 200 for γ in the objective function.

Figure 4.5 shows the transformation functions which have been produced from the product structure shown in Figure 4.4. On the path from requirement A to the component types 6-12, there is a link between configuration options 2 and 3 at requirement B. There is also a link between configuration options 4 and 5 at requirement C. Table 4.5 shows the component types and the facilities in which they can be produced or repaired. The parameters of capacity and load are also listed for each facility.

Table 4.2 shows the sequence of stages for the structure in Figure 4.4 as determined by the rules discussed earlier. According to Table 4.2, stage H can be filled by the combined component types 15/17, 15/18, 16/17 and 16/18. To determine the impact of choosing one of these component types on the MTBF of the system, we use equations (4.10) through (4.13). For the backwards recursion, we begin at stage H which brances to terminal nodes and so falls under Case I discussed earlier. We then build decision rules using equations (4.28) and (4.29) in the following manner;

if $LCC_H(15/17) - LCC_H(15/18) > 200a_k(A_H(15/17) - A_H(15/18))$ then choose 15/18 else choose 15/17



* where a_r* is either a_E or a_F

Figure 4.5: Transformation Functions for MTBF

using the information from Tables 4.3 and 4.4, the above calculation becomes;

if 33.153 - 29.725 > $200a_k(A_H(15/17) - A_H(15/18))$, choose 15/18, else choose 15/17

if 0.0171 > $a_k[A_H(15/17) - A_H(15/18)]$ then choose 15/18, else choose 15/17 =

if 0.0171 > $a_k[(.975) - (.954)]$ then choose 15/18 else choose 15/17 =

if 0.843 > a_k then choose 15/18 else choose 15/17.

To complete the decision rules for stage H, we must perform the pairwise comparisons with the other configuration options. Using the same equations as above, we will compare option 15/18 with option 16/17 and option 16/17 with 16/18. We then proceed to stage G, which also falls under case I, and build the decision rule for component types 13 and 14 in a similar fashion. Table 4.7 summarizes the decision rules which were built for each stage using the above logic.

Product Design Recursion

Once the decision rules have been created for each stage, we then use the initial values for the state variables and begin the process of making decisions. Requirement A is the first stage in the sequence, however, there is no decision to be made at this point so we move to requirement B. We will use the initial value of $a_B=1$, $g_B=0$ and the initial value of the other elements of the state vector to start building up the *LCC* and availability for the system.

Using the decision rules for requirement B from Table 4.7, we find that we will choose option 3 over option 2. We then update the transformation functions using the initial values of the state variables;

Table 4.7
Decision Rules

Stage	Option	Case	Decision Rule	$A^*(o)$	$LCC^*(o)$
H	15/17 - 15/18	I	if $a_k > 0.843$ choose 15/17	0.975	33.153
H	15/18 - 16/17	I	if $0.843 > a_k > 0.741$ choose 15/18	0.954	29.725
H	16/17 - 16/18	I	if $0.741 > a_k > 0.349$ choose 16/17	0.941	27.807
H	16/17 - 16/18	I	if $0.349 > a_k$ choose 16/18	0.924	26.585
G	14 - 13	I	if $a_k > 0.825$ choose 14	0.987	35.486
G	14 - 13	I	if $0.825 > a_k$ choose 13	0.963	31.395
C	4 - 5	II	if $a_k > 0.917$ choose 4	0.987	35.486
C	4 - 5	II	if $0.917 > a_k > 0.843$ choose 5	0.975	33.153
C	4 - 5	II	if $0.741 > a_k > 0.843$ choose 5	0.954	29.725
C	4 - 5	II	if $0.349 > a_k > 0.741$ choose 5	0.941	27.807
C	4 - 5	II	if $0.349 > a_k$ choose 5	0.924	26.585
F	11 - 12	III	if $a_k > 0.986$ choose 12	0.918	45.596
F	11 - 12	III	if $.906 < a_k < .984$ choose 12	0.907	43.263
F	11 - 12	III	if $0.797 > a_k > 0.906$ choose 12	0.888	39.834
F	11 - 12	III	if $0.375 > a_k > 0.797$ choose 12	0.876	37.916
F	11 - 12	III	if $0.375 > a_k$ choose 12	0.860	36.694
E	9 - 8	III	if $a_k > 0.98$ choose 9	0.924	59.234
E	9 - 8	III	if $0.966 < a_k < 0.98$ choose 9	0.912	56.902
E	9 - 8	III	if $0.922 < a_k < 0.966$ choose 8	0.817	52.875
E	9 - 8	III	if $0.811 < a_k < 0.922$ choose 8	0.872	49.447
E	9 - 8	III	if $0.381 < a_k < 0.811$ choose 8	0.860	47.528
E	9 - 8	III	if $0.381 > a_k$ choose 8	0.845	46.306
D	6 -7	III	if $a_k > 0.887$ choose 7	0.740	71.287
D	6 - 7	III	if $0.408 < a_k < 0.887$ choose 7	0.730	69.369
D	6 - 7	III	if $0.457 > a_k$ choose 7	0.721	68.147
B	2 - 3	II	if $A_s > 0.986$ choose 3	0.918	45.565

$a_B = 1$

$ML_3 = 0, RL_3 = 0$

$d_{1,B} = 800 - 0 = 800$

$d_{6,B} = 1500 - 0 = 1500$

$g_B = 0$

Requirements, such as B, which do not branch to terminal nodes act like 'dummy' stages which exist only to preserve the flow of the state variable information. Therefore, $LCC_r(o)$ for these types of requirements have values of zero. Given that we have chosen option 3 and that $a_B = 1$, we can determine from Table 4.7 that we should choose option 12 to fill requirement F, and;

$a_F = (.931)\ a_B = .931$

$ML_{12} = 69, RL_{12} = 111$

$d_{1,F} = 800 - 69 = 731$

$d_{6,F} = 1500 - 111 = 1389$

$g_F = 10.109 + g_B = 10.109$

Using Table 4.7 to find the decision rules, we would choose option 4 over option 5 to fill requirement C because the availability computed at stage F is greater than 0.917. Once again, requirement C serves as a dummy node with values of zero for the element **g** and 1 for the element **a**. We update the transformation functions;

$a_C = (1)a_F = 0.931$

$ML_4 = 0, RL_4 = 0$

$d_{4,C} = 1700 - 0 = 1700$

$d_{7,C} = 2000 - 0 = 2000$

$g_C = 0 + g_F = 10.109$

Once again, given that we have chosen option 4 and that $a_C = 0.931$, we would choose option 14 to fill requirement G. We then update the transformation functions again;

$$a_G = (.987)a_C = 0.918$$

$$L_{14} = 197,\ RL_{14} = 337$$

$$d_{4,G} = 1700 - 65 = 1635$$

$$d_{7,G} = 2000 - 107 = 1893$$

$$g_H = 35.486 + g_C = 45.596$$

The objective function value for this set of options is 45.596 + 200(1 - 0.918) = 61.856.

The vector $\mathbf{d}_{n,r}$ has tracked the available capacity at the production facilities and the vector $\mathbf{d}_{m,r}$ has tracked to available capacity for the repair facilities. An additional check must be made to determine whether or not there is enough capacity at a specific facility to accept the load from option *o*, before it can be included in the product design. If there is not enough capacity, then option *o* can not be considered to fill requirement *r*.

This completes the inner product design DP of equation (4.6) for the first iteration of the middle problem given a fill rate of 0.8 for each echelon from the outer problem. The next step would be to change the production facility for LRU C to be facility 5 and rework the inner problem DP. After this has been done we change the production facility for LRU B, until all the facilities have been enumerated. We then change the fill rate from the outer problem and start the process again.

At this point, we have identified three major stages within the overall optimization problem along the phases of the product life cycle. There is a further breakdown within these stages

corresponding to the decision points within the product structure. We have also identified the vector of state variables which must be maintained to make the decisions at each stage.

While Dynamic Programming will achieve the optimal solution through either the forwards recursion or the backwards recursion method, we will use the backwards approach to generate the solution to this problem. It has already been noted that the availability calculations require a 'bottom-up' approach starting from the lowest level components to compute the top level system reliability. The backwards recursion will facilitate the computation of this performance measure.

Chapter 5: Model Implementation

Chapters 3 and 4 discussed the solution approach to the product design/manugistics-system design problem outlined previously in the dissertation. The input parameters required by this solution technique, as well as the performance measures and the decision variables to be computed were also presented in Chapter 3. The modeling technique, which uses a dynamic programming framework to solve the problem, was outlined in Chapter 4. This chapter discusses the computer model which is based on the dynamic programming formulation presented in Chapter 4. The model parameters were organized into an input file for the computer program, and an example corresponding to Figure 4.4 is shown in Appendix A.1 while the field descriptions are in Appendix A.2. The FORTRAN code of the solution approach is shown in Appendix B.

Program Logic

Parameter Database

Because of the large number of parameters that are associated with the components and sub-assemblies of the product, an extensive database must be maintained to track and record each of

these parameters. The model input file serves as the means for initializing the parameter values. It contains 25 different types of records that correspond to the various parameters for the components, sub-assemblies, production system and operations support system.

Record 1 contains all of the scenario information pertaining to the production and operations processes. It provides the information concerning the number of LRU's, components, repair and operating locations, the number of production cells and the number of acquisitions for purchasing the product. Records 3, 6 and 15 store all of the component information such as, failure rate, setup time, production run time, component test time and remove-replace time. Record 5 shows the production system information. It contains setup cost, cell capacity and current production load from existing products as well as holding cost rate and labor hour rates. Records 8, 13 and 21 provide the model with all of the repair location information. They contain transportation times, transportation costs, repair facility capacities and current repair loads from existing products, ordering costs and inventory holding cost rates.

Records 9 through 12 and 16 through 18 contain all of the product structure information which is required to build the DP stages for the inner product problem of (4.6). These records describe the configuration requirements and options to depict how they are linked together in the product-structure network. These record types also indicate the type of configuration in which the requirements are grouped, such as parallel, series or standby, which controls the computation of system level availability.

Subroutines

The logic flow of the FORTRAN code follows the performance measure hierarchy chart shown in Figure 3.4. Several intermediate performance measures are shown in Figure 3.4 to be subordinate to fleet-wide system-level availability. These intermediate measures must be computed in the sequence specified in Figure 3.4 in order to calculate availability for each component option. The elements of *LCC*, from (4.8), are computed separately as the information required by the calculations becomes available from the intermediate measures. For example, the repair cost element of *LCC* can not be calculated until repair load is known. Once *LCC* and fleet-wide availability are determined for each component, the decision rules for the inner problem of (4.6) can be formulated.

The basis for most of the *LCC* calculations in Chapter 3 is the demand rate for each component or LRU. Repair cost, production cost, disposal cost, transportation cost and inventory cost all depend on the demand rate. The value of h, that was discussed in Chapter 3, must be determined for each component before expected number of demands can be computed. The fleet-wide system availability computation requires the component failure rate augmented by it's associated value of h. As mentioned in Chapter 3, the value of h for a particular component is determined by the configurations of the options in the product structure above that component.

Figure 5.1 shows the general system level flow chart for the computer code in Appendix B. After reading the model input file information and storing the data in the parameter database, the program calls on two routines which assist in building the links between the stages. This is a

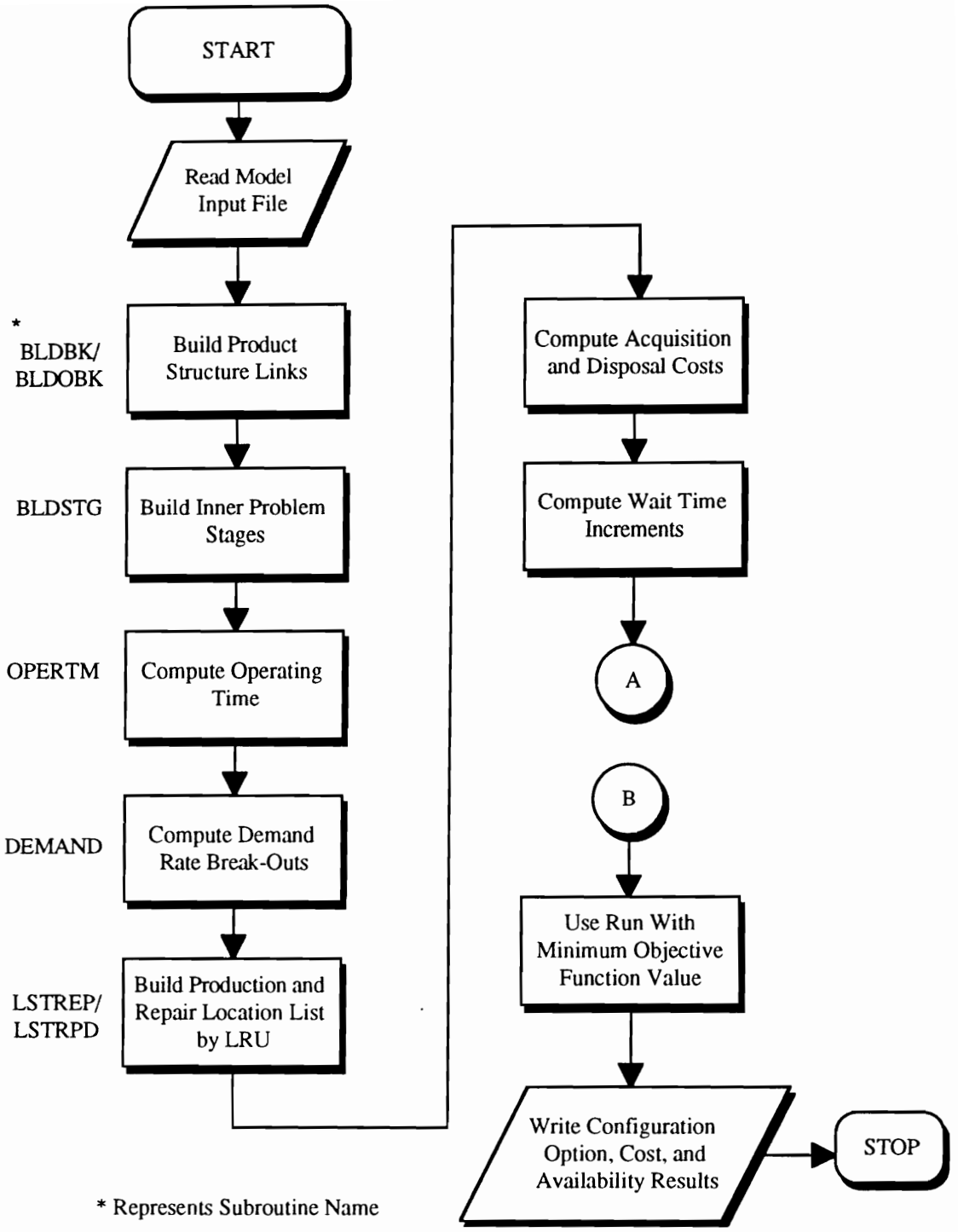


Figure 5.1: System Level Flowchart for FORTRAN Code (Page 1 of 2)

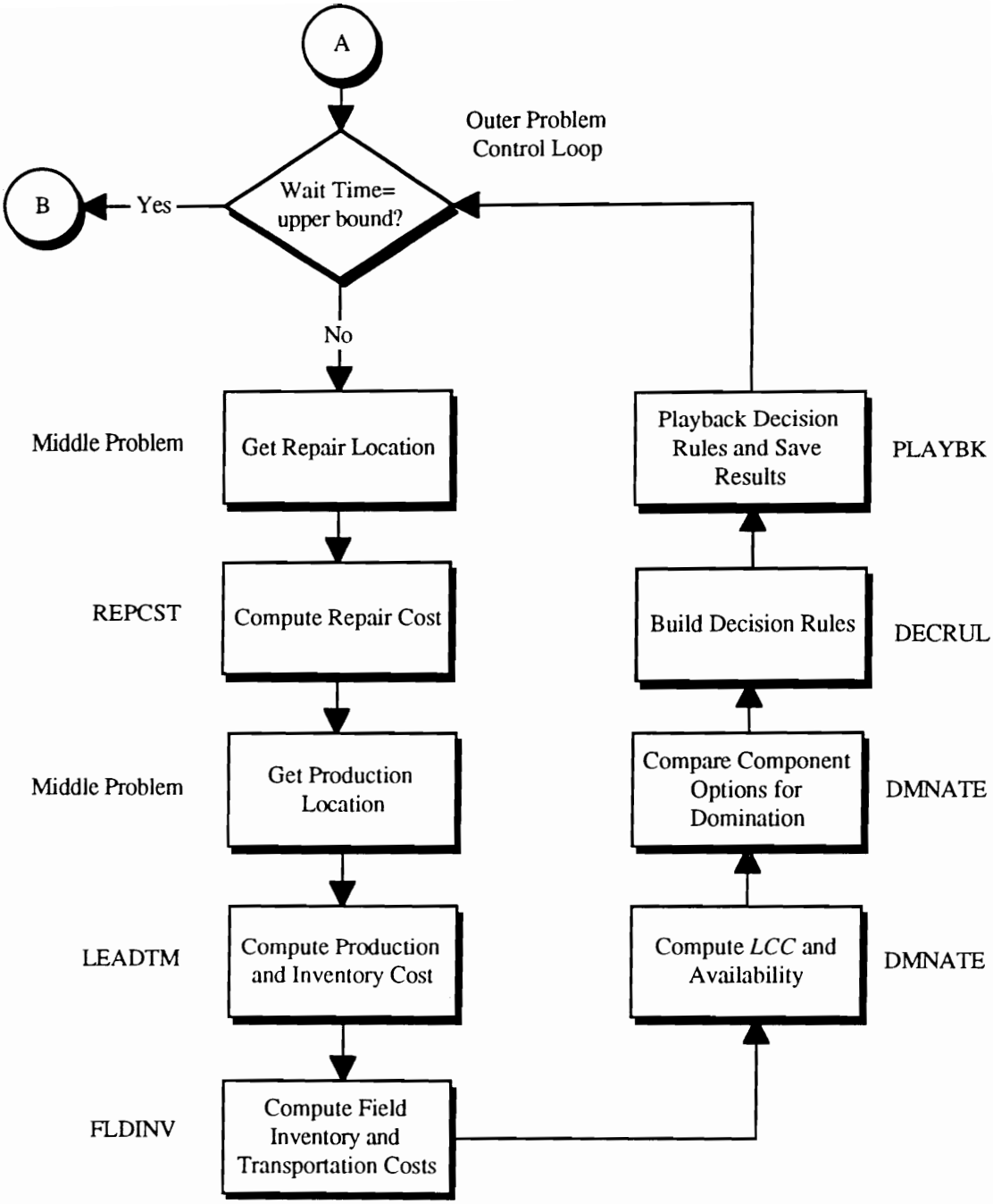


Figure 5.1: System Level Flowchart for FORTRAN Code (Page 2 of 2)

first step in computing the value of h for each component. Subroutine BLDBK finds the requirement from which each configuration and component option branches, while subroutine BLDOBK finds the previous configuration option for each requirement. These two routines provide the inputs for building the stages of the inner product DP of (4.6).

Once the links between the configuration options and the requirements have been established, the next set of subroutines compute the value of h for each component option. Subroutine BLDDH runs through each path of the product structure and determines whether or not a particular requirement is a component requirement. If the requirement is a component requirement, BLDDH sets h to a value of one, otherwise, h is set to a value of zero. Subroutine BLDTH then travels through each path of the product structure and computes a multiplier for h. This multiplier is set to a value of one for each configuration option which has a series or passive redundant configuration while it is set to a value of $(\sum_{i=1}^{p} \frac{1}{i})^{-1}$ for each option in active redundancy. The value from BLDDH is then multiplied by the value from BLDTH for each requirement.

The next routine builds the stages for the inner product problem of (4.6) using the rules outlined in Chapter 4. Subroutine BLDSTG travels through each path of the product structure to determine the ordering of the inner problem stages. Each stage of the inner problem DP corresponds to a requirement in the product structure. Once the building of the stages has been accomplished, the value of h that is associated with each stage that represents a component requirement is multiplied by the component failure rates for that particular requirement. This augmented failure rate represents the component's contribution to the system level failure rate.

The BLDSTG routine also calls on the STNDBY routine whenever a requirement in the product structure represents a stage that is embedded in a passive redundant configuration option. As explained in Chapter 4, each option of a stand-by assembly will be enumerated and the entire assembly is considered to be one stage within the inner problem DP framework. Combining the stand-by assemblies into one stage in this manner facilitates the fleet-wide system level availability calculation later in the program.

In order to determine the demand rate at each location, the operating time per location and operating time ratio per requirement of the product must first be computed. Subroutine OPRATE travels through each path of the product structure and brings the operating time ratio of each intermediate requirement forward to it's subordinate requirements. The subroutine OPERTM computes the total operating time for the product by location subject to the number of systems purchased during each acquisition phase. The operating time by location is then augmented by the operating time ratio of each requirement.

The next two sets of routines compute the demand rate for each component option and stand-by enumeration for the product. The demand rates are computed to serve as inputs for several of the *LCC* calculations. Demand rates are calculated for each operating and repair location by component option, for the manufacturing location, by LRU at each operating and repair location and finally for each LRU by component. Demand rates for component are computed in this manner to compute the order quantities and reorder points at each operating and repair location as well as the production quantities for the manufacturer. The number of demands at each repair facility is the input for the repair and transportation cost calculations by location.

Two of the inputs required by the program are the set of production locations for each component and the set of repair locations for each LRU. Subroutine LSTREP reads the set of repair locations and builds an array which represents the list of location choices for the set of LRU's. Subroutine LSTPRD performs the same set of operations for each component by production location. This routine provides the combination of repair and production locations that are needed for the middle optimization problem of (4.6).

Since acquisition and disposal costs do not depend on a particular production or repair location, they are computed for each component prior to the inner problem optimization. The outer problem of (4.6) sets as state variables for the inner problem the average wait time while the middle problem sets the repair and production facility choices as state information. Once the inner problem has this information, it begins the optimization process by computing the other elements of *LCC* and fleet-wide system availability.

The first subroutine of the inner problem uses component demand by repair location as an input to calculate the repair load. Subroutine REPCST uses this information along with the repair load information to compute the repair cost for each component. Subroutine STNREP performs the same set of operations for the standby assemblies.

The next subroutine of the inner problem computes production load for each component in the product structure network. As shown in Figure 3.4, this measure is needed as an input to computing manufacturing leadtime. Subroutine LEADTM accepts average wait time and component demand as inputs for subroutine HW. HW is a routine based on the Hadley-Whitin

method of optimizing production quantity and reorder point to meet the average wait time criterion. HW returns the optimal values of production quantity and reorder point for each component and LEADTM uses the setup time and cycle time for the production cell that was sent by the middle problem and computes manufacturing leadtime as well as production cost.

Subroutine FLDINV computes the inventory cost and transportation cost for each operating and repair location in the manugistics system. It also uses the HW subroutine to compute order quantity, reorder point and safety stock for the components at each location. Since the repair location condemns and places orders for LRU's and components, FLDINV requires as input, the repair location choice from the middle problem. If the repair choice is the operating location, the transportation time is from the manufacturer to the operating location. If the repair choice is a depot, then the transportation time is from the manufacturer to the depot, and then to the operating location.

After calculating the LCC and availability for each component, the program calls subroutine DMNATE. This routine evaluates the options for each component requirement to determine if any choice dominates any other choice. A particular option is said to dominate another option if it has a higher availability at a lower *LCC*. Subroutine DMNATE compares each option and sets a flag if an option has been dominated. This flag prevents the option from being considered as a feasible choice for a component requirement later in the program.

Subroutine DECRUL builds the decision rules for each stage based on *LCC* and average fleet-wide system availability. As mentioned in Chapter 4, there are three cases of decision rules

which depend on the type of requirement a particular stage represents. The decision rule calculation associated with Cases I and II is a special form of the general computation used for Case III. Subroutine DECRUL begins with stage 1 and travels through the following stages to determine which case must be used to build the decision rule for that stage. The subroutine then stores each decision point in a array for later use.

The last subroutine of the inner problem travels through the stages in the reverse direction to evaluate the decision rules and decide which configuration and component options should be used for each configuration requirement. Subroutine PLAYBK uses the array of decision points to make the decision at each stage and update the state information for the following stage. It also computes a final value of the objective function, $LCC + \gamma U$, and stores it with the choices of repair and production facilities, order and production quantities, safety stocks and component options.

After the subroutines of the inner problem have completed their calculations, the middle problem chooses a new set of production and repair facilities for each LRU and provides this as an update to the state vector. The inner problem then repeats its calculations. This is done until all of the sets of repair and production locations have been enumerated by the middle problem. The last set of operations performed by the program evaluate all of the objective function values stored by subroutine PLAYBK and determines which iteration has the lowest value. The configuration options choices, order and production quantities, repair and production locations, LCC, fleet-wide system availability and objective function values are then written to an output file.

Program Results

Several problems of various sizes were generated to evaluate the effect of problem size on total run time for this particular solution approach. The factors that determine the size of a problem are: number of LRU's, number of stages, number of production facility choices per LRU, number of repair location choices per LRU and number of configuration options per configuration requirement. The number of LRU's represents the width of the product structure network while number of stages corresponds to the depth of the network. The number of configuration options will affect the complexity of the decision rules that must be tracked for the PLAYBK subroutine. Finally, number of LRU's and number of production and repair locations per LRU directly affect the number of iterations for the inner optimization problem.

Table 5.1A shows the different levels for each factor that was used to test the run time for small, medium and large number of LRU's. As the number of LRU's and associated combinations of production and repair choices per LRU increases, the number of iterations for the inner problem of (4.6) increases. Table 5.1B shows how these combinations were limited in the case of a large number of LRU's. The small case allows four LRU's to have two production and repair choices with the remaining three having one choice. The medium case allowed for 3 LRU's to have three choices, 2 to have two choices and 2 to have 1 choice. The large case allowed for all 7 LRU's to have two choices. These modified levels are not unreasonable even for large manugistics systems.

Table 5.1A

Problem Size Factors

Factor	Small	Medium	Large
Number of LRU's	2	5	7
Number of Production Cells	1	3	5
Number of Repair Locations	1	3	5
Number of Options per Requirement	1	2	3
Number of Stages	10	15	20

Table 5.1B

Large LRU Case Factors

Factor	Small	Medium	Large
Number of Production-Cell choices per LRU	4 LRUs, 2 choices 3 LRUs, 1 choice	2 LRUs, 3 choices 2 LRUs, 2 choices 3 LRUs, 1 choice	All LRU's have 2 choices
Number of Repair-Location choices per LRU	4 LRUs, 2 choices 3 LRUs, 1 choice	2 LRUs, 3 choices 2 LRUs, 2 choices 3 LRUs, 1 choice	All LRU's have 2 choices

Tables 5.2A, 5.2B and 5.2C show the combination of factor levels and the resulting run times for the 63 runs used in the design of the test. The value of *S* corresponds to the small level, the value of *M* represents the medium level and the value of *L* is the large level. The runs were made on an IBM 3090 model 300E/VF mainframe system and the run time values are in CPU seconds. The run times for the small LRU case problems are quite reasonable with the largest time being 1168.06 CPU seconds. The medium LRU case problems represent a fairly complex product structure. The longest run time, for the medium level of production and repair choices per LRU is 2196.89 CPU seconds. We state once again, even for large manugistics systems, the number of repair choices would not exceed 2 or 3 choices per LRU.

The run time for this program is directly related to the number of iterations that the inner optimization must perform to solve the problem. For this dissertation, an enumeration scheme for the outer and middle optimizations was chosen. This rudimentary method for these optimizations is the principle cause of the increase in CPU time with problem size.

Although the run times for the larger LRU cases are long, this model was designed as a tool for use during the design stage of the product life cycle. Consequently, the model will not be run frequently and for most major design efforts, the cost for run time would only be a fraction of the total research and development cost. The model evaluates all production, repair, component and configuration alternatives associated with a particular product design effort. This 'single run' approach reduces the number of times that model runs are generated and also reduces the number of different scenarios that are required as input data by the model.

Table 5.2A
Factor Levels and Run Times, Small LRU Case

Number of LRUs	Number of Stages	Number of Production Cells	Number of Repair Locations	Number of Options	Run Time (Seconds)
S	S	S	S	S	1.47
S	S	S	M	L	8.35
S	S	S	L	M	27.61
S	S	M	S	L	8.56
S	S	M	M	M	49.74
S	S	M	L	S	165.02
S	S	L	S	M	28.04
S	S	L	M	S	165.54
S	S	L	L	L	552.33
S	M	S	S	M	2.28
S	M	S	M	S	13.27
S	M	S	L	L	43.33
S	M	M	S	S	13.49
S	M	M	M	L	79.13
S	M	M	L	M	259.94
S	M	L	S	L	44.25
S	M	L	M	M	263.21
S	M	L	L	S	867.16
S	L	S	S	L	3.08
S	L	S	M	M	17.89
S	L	S	L	S	58.80
S	L	M	S	M	18.40
S	L	M	M	S	107.53
S	L	M	L	L	354.81
S	L	L	S	S	60.26
S	L	L	M	L	359.01
S	L	L	L	M	1168.06

Table 5.2B

Factor Levels and Run Times, Medium LRU Case

Number of LRUs	Number of Stages	Number of Production Cells	Number of Repair Locations	Number of Options	Run Time (Seconds)
M	S	S	S	L	1.25
M	S	S	M	M	37.18
M	S	S	L	S	284.83
M	S	M	S	M	38.39
M	S	M	M	S	1189.38
M	S	L	S	S	292.12
M	M	S	S	S	1.76
M	M	S	M	L	53.19
M	M	S	L	M	399.72
M	M	M	S	L	54.21
M	M	M	M	M	1672.02
M	M	L	S	M	414.81
M	L	S	S	M	2.16
M	L	S	M	S	66.49
M	L	S	L	L	519.64
M	L	M	S	S	71.01
M	L	M	M	L	2196.89
M	L	L	S	L	547.54

Table 5.2C

Factor Levels and Run Times, Large LRU Case

Number of LRUs	Number of Stages	Number of Production Cells	Number of Repair Locations	Number of Options	Run Time (Seconds)
L	S	S	S	M	329.26
L	S	S	M	S	743.92
L	S	S	L	L	2631.68
L	S	M	S	S	759.08
L	S	M	M	L	1671.84
L	S	L	S	L	2659.82
L	M	S	S	L	429.64
L	M	S	M	M	95.65
L	M	S	L	S	3367.68
L	M	M	S	M	979.21
L	M	M	M	S	2139.95
L	M	L	S	S	3404.38
L	L	S	S	S	515.56
L	L	S	M	S	1180.36
L	L	S	L	L	4074.89
L	L	M	S	M	1184.84
L	L	M	M	L	2589.33
L	L	L	S	M	4153.34

Evaluation of Objective Function

This section illustrates the effect of gamma, the marginal exchange rate in the objective function, $LCC + \gamma U$, on the decision rules and the final choices of configuration options for the requirements in the product structure. The value of gamma in the objective function serves as a weighting factor for the tradeoff between *LCC* and availability. This is the basis for comparing component options to determine if any option is dominated by another one. It also affects the endpoint that is computed by the decision rules. Tables 5.3A and 5.3B present a small example of 3 configuration options for one particular requirement of a given product. Table 5.3A contains the values of availability and *LCC* that will be used in the decision rule formulation while Table 5.3B shows the computed decision rules for each option comparison.

For this illustration, the values of gamma range from a low of 40 to a high of 150. The first set of decision rules are based on the value of 40 for gamma. This lower value of gamma allows for more emphasis to be placed on *LCC* than system availability. As the decision rule shows, configuration option 3 will always be chosen because it has the lowest value for *LCC*. As the value of gamma increases, more weight is placed on the availability portion of the objective function and the decision rules reflect this by choosing the option with the highest availability. For example, when gamma is 150, the last set of decision rules indicate that option 1 should be chosen when the incoming availability if greater than 0.325. The range for choosing option 1, which has the greatest availability, increases as the value of gamma increases.

Table 5.3A
Option, Availability and LCC

Option	Availability	LCC
1	0.9541	5.14108
2	0.9290	3.91409
3	0.9113	3.15784

Table 5.3B
Gamma Versus Decision Rules

Gamma	Comparison	Endpoint	Rule
40	1 - 2	1.221	always choose 3
	2 - 3	1.068	
50	1 - 2	0.976	if a > 0.976 choose 1
	2 - 3	0.854	if 0.854 < a < 0.976 choose 2
			if 0.0 < a < 0.854 choose 3
60	1 - 2	0.814	if a > 0.814 choose 1
	2 - 3	0.712	if 0.712 < a < 0.814 choose 2
			if 0.0 < a < 0.712 choose 3
70	1 - 2	0.697	if a > 0.697 choose 1
	2 - 3	0.611	if 0.697 < a < 0.611 choose 2
			if 0.0 < a < 0.611 choose 3
80	1 - 2	0.611	if a > 0.611 choose 1
	2 - 3	0.534	if 0.534 < a < 0.611 choose 2
			if 0.0 < a < 0.534 choose 3
100	1 - 2	0.488	if a > 0.488 choose 1
	2 - 3	0.427	if 0.427 < a < 0.488 choose 2
			if 0.0 < a < 0.427 choose 3
150	1 - 2	0.325	if a > 0.325 choose 1
	2 - 3	0.284	if 0.284 < a < 0.325 choose 2
			if 0.0 < a < 0.284 choose 3

Sensitivity Analysis

A sensitivity analysis was conducted for several of the input parameters to determine the impact their values have on the final result. There were two reasons for performing this sensitivity analysis; first, to illustrate the complexity of some of the tradeoffs in the design problem and thereby demonstrate the need for this optimization model and second, to illustrate some instances where solutions are sensitive or insensitive to certain parameters, thereby illustrating how the model can help direct parameter estimation efforts.

Each parameter that was included in this analysis was changed by a factor of 2, 5 and 10 over the original value. Table 5.4 shows, for the factor of 10 case, the summary measures of *LCC* and fleet-wide system level availability for the product as well as the input parameter that was changed. The scenario that was used as baseline consists of 5 LRU's, 10 stages, 3 repair and production location choices and 2 options per requirement. This scenario is represented in Table 5.4 as run number one.

Repair labor cost rate, $C2_n$, had no affect on availability because the same set of configuration options were chosen to fill the requirements. For the same reasons, changing the repair times also had no affect on the final results. Total repair cost, *RC*, is based on the mean time to repair an LRU, the repair labor rate by location and the assignment of LRU's to repair location. Given that repair time stays constant, choice of repair location then becomes an issue of comparing the labor costs. Consequently, for this example, the product designers would be ill-advised to spend further effort in estimating more accurately the labor cost rates and repair times.

Table 5.4

Sensitivity Analysis

Run Number	Evaluted Parameter	Availabilty	LCC
1	Baseline	0.6144	33.162
2	Repair Labor Cost	0.6144	260.70
3	Cell Labor Cost	0.6133	38.358
4	Cell Order Cost	0.5607	33.672
5	Acquisition Cost	0.5440	47.349
6	Repair Order Cost	0.6144	33.912
7	Transport Cost	0.5655	56.349
8	Disposal Cost	0.6144	34.971
9	Operating Time	0.5491	62.959
10	Transport Time	0.5182	33.792
11	Repair Time	0.6144	248.32
12	Production Time	0.5475	38.369
13	Failure Rate	0.3164	61.942

Total production cost, PC, is computed from the setup cost, $C6_{jp}$, and the run time cost, $C3_p$. Setup cost is entered for each component/cell combination while labor cost is specific to each particular cell. If the run times and batch size remain constant, changing the cell labor rate will affect the run time cost portion of production cost. If this change is large enough, a cell which was previously the most economical may no longer be the best choice. For the example problem, the change in system availability occurs because one LRU did change cell choices. This affects the manufacturing leadtime, which also impacts total down time. This illustrates the value of the model in determining the impact that a manugistics system design change has on the performance of the product.

Experiment number 12 from Table 5.4, changed the production run times for the various component/cell combinations. This can impact the manufacturing leadtime which accounts for the difference in system availability from this experiment. Once again, the run times were changed by a factor of 2, 5 and 10 times the original value. Changing these times could affect the production batch size and reorder point, which would then alter the production and inventory costs, however, for these set of runs, there was no change in production cell choice or configuration option choice. The difference in LCC is a result of the change in production and inventory costs to the manufacturer.

Changing the cell ordering cost, $C6_{jp}$, can affect the production quantity at the manufacturing location that is computed by the HW subroutine, as described above. The same holds true for the repair ordering cost, $C12_n$, and the order quantity that is computed for each location. The ordering cost per location was changed by a factor of 2, 5 and 10 over the original value for

these experiments. Potentially, this can impact the manufacturing leadtime and consequently, system down time. While changing these two ordering costs had no affect on the option choices for this example, the system availability did change because of the change in leadtime and downtime.

Acquisition cost, $AC(i)$, is unique for each component and LRU. Total acquisition cost depends only on the cost per component ,$C9_j(i)$, or LRU, $C10_t(i)$, the number of acquisitions over the product life cycle and the number of systems purchased for each acquisition. If the acquisition costs of components are misspecified, the computed cost for each component that is used by the DECRUL subroutine are affected, which could impact the choices made by the PLAYBK routine. For this example, two requirements were filled by different configuration options which accounts for the chance in availability.

Total transportation cost, XC, depends on the repair location choice, the number of trips taken and the cost per trip, $C7_{nm}$. Run 7, from Table 5.4, changed the transportation cost per trip for each repair and operating location by a factor of 2, 5, and 10 over the original value. Changing the cost per trip did not affect the repair location choice for each LRU, but it did have an impact on the options that were chosen for the requirements. The overall change in cost per component was enough to impact the decision rules for the value of gamma (20) used in the example. Run number 10 changed the transportation time, T_{nm}, for each location in the example scenario. This change did affect the repair location choice for one of the LRU's. Transportation time can also impact the system down time and the system availability.

Disposal cost, DC, per LRU depends only on the condemnation rate, K_t, the disposal cost per LRU, $C8_t$, and the expected number of failures per LRU. As in the acquisition cost case, changing the disposal cost will change the computed costs which are used in the DECRUL subroutine and could potentially change the choices that are made in the PLAYBK routine. In this example, however, there was not enough of an increase to change the option, repair or production choices since disposal cost was a small portion of total LCC.

Changing the operating time, O_n, will affect every cost function that depends on expected number of demands. However, since all costs are affected in a similar fashion, changing the operating time did not affect the option, repair or production choices that were made by the program. Experiment 9 changed the operating time by a factor of 2 and 5 over the original value. Changing the failure rate, λ_j, will change the availability for each component and the entire product. This change will also affect every cost that uses expected number of demands. This run did affect one of the option choices and the system availability is lower as would be expected.

Chapter 6: Conclusions

Summary

The life cycle of any product begins with the identification of a need that is currently not being fulfilled. This phase of the life cycle is called the needs analysis stage. After the customer has communicated this need to the developer, the product enters the design stage. Upon finalizing the design, the product is manufactured, presented to the customer and operated by the customer for its remaining useful life. The final stage of the life cycle is disposal of the product. Total life cycle cost for a product is then computed by summing across all of the costs accrued at each stage, which includes acquisition, manufacturing, inventory, transportation, repair and disposal.

Most of the decisions which affect the manufacturing and logistics support structure of a product occur well before the manufacturing stage of the life cycle. These decisions are the result of trade-off analyses on performance measures such as reliability, maintainability and availability which then affect location of spare assets, number of repair personnel and other support resources. Studies have shown that almost 80% of the total support costs for a product are determined prior to production. For this reason, the focus of this dissertation research has been on the importance of product design and its impact on product performance as well as the impact on the manugistics support structure.

Product design decisions can no longer be made in a vacuum or using the sequential process that previously existed. Producing a cost effective design, that meets the needs of the customer, can only be accomplished by a design team that consists of not only the design engineers, but also the manufacturing engineers, logistics engineers and the customer. The total impact of design choices on the support process must be evaluated as early in the life cycle as possible. This requirement forms the basis for concurrent optimization of product design, facilities design and location and operations control. This type of optimization is the Operations-Research methodology required by Concurrent Engineering.

Total *LCC* and fleet-wide system availability were chosen as the two measures to represent product and manugistics system performance. These measures coincides with Garvin's [1984] value-based definition of quality which states that product quality means required performance at an acceptable price or acceptable performance at a given cost. This definition acknowledges the trade-off between system availability and *LCC*.

Research Objectives

The goal of this research is to produce a decision support model that assists the design team by evaluating the impact of alternate product designs on the manugistics support process. While most diagrams of the product life cycle represent the stages of the life cycle as distinct phases, in reality they will interact and quite often, overlap. It then becomes necessary to quantify the links between the stages of the life cycle to assess the influence a particular design will have. Specifically, this research addressed the links among product configuration, component choice,

and production costs, support costs, manugistics system design, operational control policies and system availability. The model accepts design information as provided by the design team as input and computes system level operational availability, manufacturing and logistics support costs for the expected lifetime of the product.

Concurrent optimization addresses all of the decision variables to be considered at each stage of the life cycle and determines their optimal values in a simultaneous fashion. The formulation in Chapter 3 divided the decision variables into three major categories; product design process, manugistics system design process and operations (both production and field) control support process. The model presented in Chapter 4 is based on a modular design which incorporates different embedded descriptive models that are used to compute the performance measures which determine the optimal values of the decision variables.

The contributions of this modeling approach are outlined as follows;

1) A model is constructed which links together the product design, manufacturing and logistics system design and manufacturing and field operations phases of the life cycle.

2) An optimization scheme is applied to the overall model structure to concurrently optimize the objective function criteria.

3) The final solution computed by the model is based on a multi-criteria value function formed from the individual objective functions of minimizing LCC and maximizing availability.

4) The model formulation is somewhat robust with respect to the embedded descriptive models. That is, the submodels used in the formulation can be exchanged for other models that may represent more accurately particular aspects of the system, such as inventory costs.

Solution Approach

The natural separation of the decision variables along the phases of the product life cycle allowed for a Dynamic Programming approach to be applied as the solution technique. The order in which the stages are evaluated by the D.P is shown in (4.6). The inner problem of (4.6) is further broken into sub-stages which correspond to each decision point, or requirement, of the product structure network. The constraints of the problem are the number of configuration options per requirement and the number of repair and production location choices per LRU.

The objective function uses the two measures of LCC and system availability which are conflicting in nature. Availability is increased when reliability increases or when system down time decreases. Highly reliable components generally are more expensive to produce or purchase thereby increasing the production cost element of *LCC*. System down time can be decreased by either improving repair turn time or maintaining higher levels of safety stock at the operating locations which again increases *LCC*. The two criteria of LCC and unavailability are combined into a single objective function, $LCC + \gamma U$, where U is the system unavailability and γ is the marginal rate of substitution between *LCC* and U, subjectively specified by the decision maker based on his belief as to the worth of each measure.

Since Dynamic Programming was used as the solution technique for this problem, state information, transformation functions and decision rules were formulated to perform the optimization process. The outer problem provides average wait time as state information while the middle problem provides repair and production location as the state information. The

decision rules for the inner problem use LCC and system availability as state information. The inner problem D.P. is evaluated for each combination of state information from the outer and middle problem. The principle of optimality guarantees that an optimal solution will be achieved using this solution approach. For this problem, the optimal solution is the one which builds a product and manugistics system with the lowest LCC and highest availability for a given value of γ in the objective function.

Implementation

The code for the D.P. solution was written in standard FORTRAN 77 and runs on an IBM 3090 model 300E/VF. The program maintains an extensive database for each component and LRU configuration option in the product structure. The list of input parameters is discussed in detail in Chapter 3 and the model input file definitions are shown in Appendix A.2.

Building the initial model input file can require a certain amount of effort, especially for a large product structure. As shown in Appendix A.1, a large amount of data is needed to run the model. However, conducting subsequent runs for performing a 'what-if' analysis involves making only minor changes to the input file.

The developer typically receives a set of specifications and requirements from the customer for the new product. Designs are built which are expected to meet these product requirements. The developer needs to retrieve historical data on existing components that are included in the product design. This information may come from in-house sources or from sub-contractors if the

components are purchased elsewhere. The customer needs to inform the developer of repair location choices as well as restrictions on repair choices. The developer also needs to obtain production information from internal sources and leadtime information from external sources. Having done so, the developer is ready to build the input data file and run the model for the various design and manugistic-system design alternatives.

For products using new technology, some of the input information may not be available and will have to be estimated. A sensitivity analysis was conducted on certain parameters and was presented in Chapter 5 that illustrated how the model can guide parameter estimation and data collection efforts.

The modeling approach used in this dissertation research supports the concept of Concurrent Engineering. The information that is required by the input database spans the functional areas of product design, manufacturing and logistics support. Consequently, practitioners who wish to use this model should be operating under the CE philosophy.

Several test runs of the program were performed to evaluate the effect of problem size on run time. The number of LRU's in the product structure, the number of repair and production location choices per LRU, the number of requirements and the number of configuration options per requirement determine the size of the problem. The times for medium size problems ranged from one minute of CPU time to 37 minutes of CPU time. Theses size problems represent fairly complex systems and the run times are considered to be reasonable as this model would not be run a large number of times during the design effort. As mentioned in Chapter 5, this model was

developed for use during the product design stage of the life cycle. It is not meant to run in a production environment on a daily or weekly basis. For this reason, it is felt that the longer run times for the large case problems would not be a concern during the design effort.

Future Research

Two assumptions were made, concerning redundant configurations, to facilitate the application of a D.P. framework to this problem. The first assumption was to enumerate the choices for any standby assemblies in the product and consider the entire assembly as one stage of the D.P. The second assumption required the parallel assemblies to use the same configuration option for each requirement in active redundancy. Further research can be conducted to determine if transformation functions can be formulated which would make these assumptions unnecessary.

The calculations used by some of the embedded descriptive models in this research effort were deterministic in nature. The cost formulations are based on constant demand rates for the components. The down time calculations also use expected values for repair and production leadtimes. Additional research can be performed which introduces uncertainty into the model formulation. An embedded simulation model might be appropriate to evaluate other distributions of production and repair leadtime. The results from the modeling approach outlined in this dissertation could be used as a baseline for comparison against the results from the simulation model.

Also, a restructuring of the FORTRAN code can be accomplished to determine if the program could run in a P.C. environment without substantially increasing the run time. An evaluation of the FORTRAN code could be accomplished to determine if stage and memory are being used efficiently and correct for this.

Finally, a pre-processor might be built to facilitate the data entry for the model input file. As mentioned earlier, there is a large amount of data that must be entered into the model input file before model runs can be generated. A pre-processor which can read information from a CAD system and prepare it for entry into the model input file would decrease the amount of work that is required to run the model.

Bibliography

AFLCM 57-4, *Variable Safety Level and Aircraft Availability Model*, Ch. 16, Air Force Logistics Command, Wright Patterson AFB, OH, 1991.

Ansell, J., A. Bendell, and S. Humble, "Age Replacement Under Alternative Cost Criteria," *Management Science*, 30, 3, (1984), 358-367.

Assaf, D. and B. Levikson, "On Optimal Replacement Policies," *Management Science*, 28, 11, (1982), 439-452.

Barlow, R. E. and F. Proschan, *Statistical Theory of Reliability and Life Testing*, McArdle Press Inc, Silver Springs, MD, 1981.

Berg, M., "A Preventive Replacement Policy for Units Subject to Intermittent Demand," *Operations Research*, 32, 3, (1984), 584-595.

Blanchard, B. S., *Logistics Engineering and Management*, Prentice-Hall Inc, Englewood Cliffs, New Jersey, (1992).

Brown, M., "On the Reliability of Repairable Systems," *Operations Research*, 32, 3, (1984),607-615.

Chaharbaghi, K., "Using Simulation to Solve Design and Operational Problems," *International Journal of Operations and Production Management*, 10, 9, (1990), 89-105.

Chapman, W. L., A. T. Bahill and A. W. Wymore, *Engineering Modeling and Design*, CRC Press, boca Raton, Florida, (1992).

Clark, A. J., "An Informal Survey of Multi-Echelon Inventory Theory," *Naval Research Logistics Quarterly*, 19, 4, (1972), 621-649.

Cox, D. R., "Quality and Reliability: Some Recent Developements and a Historical Perspective," *Journal of the Operational Research Society*, 41, (1990), 95-101.

Fabrycky, W. J. and B. S. Blanchard, *Systems Engineering and Analysis*, Prentice-Hall Inc, Englewood Cliffs, New Jersey, (1990).

Fabrycky, and B. S. Blanchard, *Life Cycle Cost and Economic Analysis*, Prentice-Hall Inc, Englewood Cliffs, New Jersey, (1991).

Garvin, D.A., What Does "Product Quality" Really Mean?, *Sloan Management Review*, 26, 1, (1984), 25-43.

Geisler, M.A. and B.L. Murrie, "Assesment of Aircraft Logistics Planning Models," *Omega*, 9, 1, (1981), 59-69.

Geoffrion, A. M., "Elements of Large-Scale Mathematical Programming", *Management Science*, 16, 11, (1990), 652-691.

Graves, S. C., "Determining the Spares and Staffing Levels for a Repair Depot," *Journal of Manufacturing and Management*, 1, 2, (1988), 227-241.

Gross, D. and J. F. Ince, "Spares Provisioning for Repairable Items: Cyclic Queues in Light Traffic," *AIIE Transactions*, 10, 3, (1978), 307-314.

Hutchenson, N. E., *An Integrated Approach to Logistics Management*, Prentice-Hall Inc, Englewood Cliffs, New Jersey, (1987).

Jedrzejowicz, P., "An Overview on available models and techniques for the Multi-Criteria Reliability Problem with Emphasis on a Potential use in a DSS for this Problem Type," *Annals of Operations Research*, 16, (1988), 413-424.

Kaplan, A. and D. Orr, "An Optimum Multiechelon Repair Policy and Stockage Model," *Naval Research Logistics Quarterly*, 32, (1985), 551-566.

Lewis, E. E., *Introduction to Reliability Engineering*, Ch. 9, John Wiley and Sons, New York, NY, 1987.

Mazumdar, M., "A Comparison of Several Estimates of Availability," *Naval Research Logistics Quarterly*, 29, 3, (1982), 411-418.

Mefford, R.N., "Quality and Productivity: The Linkage," *International Journal of Production Economics*, 24, 1/2, (1991), 137-145.

Misra, K. B., and M. D. Ljubojevic, "Optimal Reliability Design of a System: A New Look," *IEEE Transactions on Reliability*, R-22, 3, (1973), 255-258.

Misterek, S.D.A., K.J. Dooley and J.C. Anderson, "Productivity as a Performance Measure," *International Journal of Operations and Production Management*, 12, 1, (1992), 29-45.

Mitten, L.G. and G.L. Nemhauser, "Multistage Optimization", *Chemical Engineering Progress*, 59, 1, (1963), 52-60.

Moore, T.P., "Optimal Design, Procurement and Support of Multiple Repairable Equipment and Logistic Systems," Unpublished Doctoral Dissertation, VPI&SU, (1986).

Murthy, D. N. P., "Optimal Reliability Choice in Product Design," *Engineering Optimization*, 15, (1990), 281-294.

Nahimas, S., "Managing Repairable Item Inventory Systems: A Review," in Multi-Level Production/Inventory Control Systems, edited by L.B. Schwarz, *TIMS Studies in the Management Sciences*, North-Holland Amsterdam, Vol 16, (1981).

Patton, J. D., *Logistics Technology and Management The New Approach*, The Solomon Press, New York, 1986.

Pierskalla, W. P. and J. A. Voelker, "A Survey of Maintenance Models: The Control and Surveillance of Deteriorating Systems," *Naval Research Logistics Quarterly*, 23, 3, (1976), 353-388.

Reasor, R. J., "Decision Support System for Integrated Design Analysis of A Repairable Item and Its Logistic Support System", Unpublished Doctoral Dissertation, VPI&SU, (1990).

Shirley, G.V., "Models for Managing the Redesign and Manufacture of Product Sets," *Journal of Manufacturing and Management*, 3, 2, (1990), 85-104.

Tapiero, C. S., P. H. Ritchken and A. Reisman, "Reliablity, Pricing and Quality Control," *European Journal of Operational Research*, 31, (1987), 37-45.

Taylor, H. M. and B. E. Rodriguez, "Optimal Replacement for Fault-Tolerant Systems," *Naval Research Logistics Quarterly*, 33, 4, (1986), 747-757.

Wisner, J.D., and S.E. Fawcett, "Linking Firm Strategy to Operating Decisions Through Performance Measurement," *Production Inventory Management*, 32, 3, (1991), 5-11.

Appendix A.1

Model Input File Record Definitions

Col	Field Definition	Type
1- 2	Record 1	Integer
3- 7	Number of production cells	Integer
8-12	Number of operating locations	Integer
13-17	Number of operating time units	Integer
18-18	Time unit indicator, 0=day,1=mo,2=yr	Integer
19-23	Number of depot locations	Integer
24-28	Number of components	Integer
29-33	Number of options - excluding component options	Integer
34-43	Gamma	Integer
44-48	Number of requirements	Integer
49-53	Number of LRU's	Integer
54-58	Number of acquisitions	Integer
59-63	Number of component options	Integer
1- 2	Record 2	Integer
3-17	Component number	Character
18-22	Component option number	Integer
1- 2	Record 3	Integer
3-17	Component number	Character
18-24	Failure rate	Real (7.6)
25-32	Purchase price	Real (8.2)
33-35	Number of production cells in which component can be manufactured	Integer
36-38	Number of LRU's on which this component is placed	Integer
1- 2	Record 4	Integer
3-17	Component number	Character
18-21	Production Cell (repeated 10 times)	Integer
1- 2	Record 5	Integer
3- 8	Cell number	Integer
9-16	Labor Cost per cell hour	Real (8.2)
17-23	Cell capacity in hours	Real (7.1)
24-30	Cell load in hours	Real (7.1)
31-36	Wait time per cell in hours	Real (6.2)
37-42	Ordering cost	Real (6.2)
43-45	Holding cost rate	Real (3.2)

Col	Field Definition	Type
1- 2	Record 6	Integer
3-17	Component Number	Character
18-23	Cell Number	Integer
24-31	Setup Cost	Real (8.2)
32-39	Cycle Time	Real (8.2)
40-47	Setup Time	Real (8.2)
1- 2	Record 7	Integer
3-12	LRU Number	Character
13-16	Condemnation Rate	Real (4.3)
17-24	Disposal Cost per LRU	Real (8.2)
25-27	Number of repair locations in which the LRU can be repaired	Integer
28-31	Number of component requirements for this LRU	Integer
32-35	Number of components for this LRU	Integer
36-39	Number of productions cells in which the LRU can be produced	Integer
1- 2	Record 8	Integer
3-12	LRU Number	Character
13-16	Repair Location Name	Character
17-20	LRU test time in hours	Real (4.1)
21-24	LRU remove and replace time in hours	Real (4.1)
1- 2	Record 9	Integer
3-12	LRU Number	Character
13-15	Component Requirement (repeated 20 times)	Character
1- 2	Record 10	Integer
3-12	LRU Number	Character
13-16	Production cell list (repeated 10 times)	Integer
1- 2	Record 11	Integer
3- 7	Option Number (non-component options)	Integer
8- 8	Option Indicator 1=series, 2=parallel, 3=standby	Integer
9-11	Number of requirements for option	Integer
1- 2	Record 12	Integer
3- 7	Option Number (non-component options)	Integer
8-10	Requirement list for option (repeated 15 times)	Character

Col	Field Definition	Type
1- 2	Record 13	Integer
3- 6	Repair location name (includes operating locations)	Character
7-14	Repair labor cost per hour	Real (8.2)
15-20	Repair location capacity in hours	Real (6.2)
21-26	Repair location load in hours	Real (6.2)
27-32	Operating time for location	Real (6.2)
33-36	Holding cost rate for components	Real (4.3)
37-40	Holding cost rate for LRU's	Real (4.3)
41-43	Number of repair depots which support this location	Integer
44-49	Ordering cost for location	Real (6.2)
1- 2	Record 14	Integer
3- 5	Echelon, 1=operating location, 2=depot, 3=manufacturer	Integer
6-11	Lower bound on wait time in days	Real (6.2)
12-17	Upper bound on wait time in days	Real (6.2)
1- 2	Record 15	Integer
3- 7	Component Option Number	Integer
8-22	Component Number	Character
23-32	LRU Number	Character
33-36	Component test time in hours per LRU	Real (4.1)
37-41	Component remove and replace time in hours per LRU	Real (5.2)
1- 2	Record 16	Integer
3- 5	Requirement	Character
6-11	Operating time ratio	Real (6.2)
12-15	Number of options which branch from requirement	Integer
16-16	Component Indicator 0=no, 1=yes	Integer
17-19	Number of requirements which are linked to this one	Integer
1- 2	Record 17	Integer
3- 5	Requirement	Character
6- 9	Option number (includes component options) (repeated 15 times)	Integer
1- 2	Record 18	Integer
3-12	LRU number (repeated 7 times) can have multiple records if more than 7 LRU's	Character
1- 2	Record 19	Integer
3- 6	Operating Location set (repeated 15 times) can have multiple records if more than 15 locations	Character

Col	Field Definition	Type
1- 2	Record 20	Integer
3- 6	Repair Location Set (includes operating locations) (repeated 15 times) can have multiple records if more than 15 locations	Integer
1- 2	Record 21	Integer
3- 6	Operating Location Name	Character
7-12	Transportation Cost	Real (6.2)
13-15	Transportation Time in days	Real (3.1)
16-19	Depot Name The last three fields are repeated 4 times, enter one set of the three fields for each depot which supports the operating location	Character
1- 2	Record 22	Integer
3- 7	Time of acquisition	Integer
8-12	Number of products purchased during acquisition	Integer
13-17	Number of locations to receive product	Integer
1- 2	Record 23	Integer
3- 7	Time of acquisition	Integer
8-12	Number of products acquired for loaction	Integer
13-16	Location name which receives product The last two fields are repeated 7 times. Can have multiple records if more than 7 locations	Character
1- 2	Record 24	Integer
3-17	Component Number	Character
18-20	Time of acquisition	Integer
21-28	Acquisition cost The last two fields are repeated 5 times. Can have multiple records if more than 5 acquistions	Real (8.2)
1- 2	Record 25	Integer
3-12	LRU number	Character
13-15	Time of acquisition	Integer
16-23	Acquisition cost	Real (8.2)
24-30	Final cost The last two fields are repeated 3 times. Can have multiple records if more than 3 acquisitions.	Real (8.2)

Appendix A.2

Example Model Input File

```
01    5     5    102     2    12     5       20     9     2     2    13
02                 6     6
02                 7     7
02                 6     8
02                 9     9
02                10    10
02                11    11
02                12    12
02                13    13
02                14    14
02                15    15
02                16    16
02                17    17
02                18    18
03                 60000909    90000   3   1
03                 70001538    65000   3   1
03                 90000767   142500   3   1
03                100000833    55000   3   1
03                110001218    72500   3   1
03                120001111   135000   3   1
03                130001531    67500   2   1
03                140001225   115000   2   1
03                150001351    67500   2   1
03                160001050    55000   2   1
03                170001176    85000   2   1
03                180002000    85000   2   1
04                 6    1    2    3
04                 7    1    2    3
04                 9    1    2    3
04                10    1    2    3
04                11    1    2    3
04                12    1    2    3
04                13    4    5
04                14    4    5
04                15    4    5
04                16    4    5
04                17    4    5
04                18    4    5
05      1     2000   20000   12000 10000 10000010
05      2     3500   30000   18000 12500 12500010
05      3     2500   25000   10000  7500  7500010
05      4     1800   34000   17000 10000 10000010
05      5     2500   15000   10000 15000 15000010
06                 6      1    12000    1500     2000
```

```
06                6      2     8000    1000     3500
06                6      3     9500    1100     2800
05                7      1     5000     570     3200
06                7      2     7500     830     2100
06                7      3    10000    1000     5000
06                9      1    15500    2900     2500
06                9      2     8500     900     2000
06                9      3     6000     850     1000
06               10      1    11000    1000     3000
06               10      2     8000    1000     3500
06               10      3    15000     750     2000
06               11      1    13000     500     3500
06               11      2    12000     750     3000
06               11      3     9000     500     3000
06               12      1    19000    1500     5500
06               12      2    11000     750     3000
06               12      3     8000     500     2000
06               13      4    11000     150     2000
06               13      5    10000     500     1000
06               14      4    13300     500     2500
06               14      5     9500     600     2000
06               15      4    15000    2000     5000
06               15      5    10000     800     3000
06               16      4    10000    1500     2500
06               16      5    17000    2000     5000
06               17      4    20000     750     6000
06               17      5    15000     500     3000
06               18      4    12000     650     4000
06               18      5    10000     900     7500
07          B0065   12500  2   3   7   3
07          C0030   15000  2   3   6   2
08          B   105000100
08          B   205000100
08          B   305000100
08          B   405000100
08          B   505000100
08          B   605000100
08          C   103000150
08          C   203000150
08          C   303000150
08          C   403000150
08          C   503000150
08          C   703000150
09          B  D  E  F
09          C  G  H  I
10          B   1   2   3
10          C   4   5
11    11  2
11    21  2
11    32  2
```

```
11     42  2
11     53  2
12     1   B   C
12     2   D   E
12     3   F   F
12     4   G   G
12     5   H   I
13    1     1500150000 80000150000010001002    10000
13    2     1200100000 40000150000000800082    12500
13    3     1000120000 60000150000015001502     9000
13    4     1000150000 80000150000010001002    10000
13    5     1300100000 40000150000000800082    15000
13    6     2000250000100000150000010001002    15000
13    7     2500200000 80000150000015001502    17500
14   1    500    900
14   2    300    700
14   3    100    500
15     6                6          B015000050
15     7                7          B018000100
15     8                6          B020000075
15     9                9          B020000125
15    10               10          B017000050
15    11               11          B025000075
15    12               12          B030000100
15    13               13          C010000125
15    14               14          C015000075
15    15               15          C020000100
15    16               16          C025000050
15    17               17          C010000100
15    18               18          C015000125
16   A    100    10
16   B    100    20
16   C    120    20
16   D    100    21
16   E    100    31
16   F    100    21
16   G    100    21
16   H    100    21
16   I    100    21
17   A    1
17   B    2    3
17   C    4    5
17   D    6    7
17   E    8    9   10
17   F   11   12
17   G   13   14
17   H   15   16
17   I   17   18
18           B          C
19    1    2    3    4    5
```

```
20    1    2    3    4    5    6    7
21    1 50000 40    6 75000 30    7 60000 20    0
21    2 60000 37    6 45000 32    7 50000 20    0
21    3 25000 10    6100000 41    7 45000 20    0
21    4 75000 30    6 50000 20    7 75000 20    0
21    5 50000 20    6 75000 30    7 60000 20    0
22     1   25     5
22     2   25     5
23     1    5    1     5    2     5    3     5    4     5    5
23     2    5    1     5    2     5    3     5    4     5    5
24                 6   1    20000   2    25000
24                 7   1    25000   2    30000
24                 9   1    50000   2    55000
24                10   1    75000   2    80000
24                11   1    45000   2    50000
24                12   1    90000   2    95000
24                13   1    80000   2    85000
24                14   1    60000   2    65000
24                15   1    50000   2    55000
24                16   1    75000   2    80000
24                17   1    95000   2   100000
24                18   1    50000   2    55000
25          B   1  200000  250000   2  250000  300000
25          C   1  300000  350000   2  350000  400000
```

Appendix B

FORTRAN Code

```
c ****
c This program runs the d.p. solution outlined in chapter 4
c ****

      INTEGER NMCELL, NUMLOC, TMUNIT, TMIND, NUMDEP, NMCOMP
      INTEGER NMOPTN, GAMMA, NMRQMT, CELCNT, NUMLRU, NMCMOP
      INTEGER NMRPLC, LRUCNT, LOCCNT, REPCNT, DEPCNT
      INTEGER ACQCNT, ACNT, ACQCT, NUMACQ,NR,RN,RUN
      INTEGER NMPROD(50), PRDCEL(50,10), CELNUM(10)
      INTEGER NUMREP(20), NMCMP(20),LACQN(10,10)
      INTEGER NMCMRQ(20), OPTN(50), OPTIND(50)
      INTEGER NUMOP(50), RQMTOP(30,5),NLINK(35)
      INTEGER DEPS(25), COMIND(25), TIME(25), STGCNT(25)
      INTEGER ACQ(25), ACQLOC(50), NACQ(25,50), ACQN(50,25)
      INTEGER OPRQMT(50),OBACK(50),LPROD(25),CHOICE(35,50)
      INTEGER LPRDCL(10,10),CNT(25),LRUPRD(5000,10),NL(50)
      INTEGER PCNT,RCNT,PDCM(50),OP,C1,PRDN(50)
      INTEGER ENM(5,10,25),OP1,C2,DSFLG(5,25),DFLG(50)
      INTEGER*4 CODEA,CODEB

      REAL LAMBDA(50), CSTPUR(50), CSTLAB(25), CELLWT(25)
      REAL CSTSET(50,7), CYCLTM(50,7), SETPTM(50,7)
      REAL CNDMRT(20), CSTDIS(20), CSTRPL(60), LOWAIT(60)
      REAL UPWAIT(60), OPTIME(60), CSTCHD(60), CSTLHD(60)
      REAL COMTST(50), OPTMRT(35), TRNCST(15,5)
      REAL TRANTM(15,5), CMACST(50,5), LRUCST(10,5)
      REAL FINCST(10,5), LRUTST(10,15), LRURPL(50,25),H(50)
      REAL REMAIN, RMNDR, REMNDR, ACREM, AREM, ACRMND
      REAL DH(25), TH(25), RATE(50), RT(10,50), SDMND(25,50)
      REAL DMNDRT(50), RATIO(35), ORATIO(35), CUMOP(50)
      REAL CDMDLC(20,50), SDMDLC(20,20,50), CSTORD(25)
      REAL TOTCDM(50), TOTSDM(25,50), CLRUDM(50,25)
      REAL FAILRT(50), CLDMLC(50,25,20), HLDCST(10)
      REAL CELCAP(25),CELOAD(25),LT(50),SLT(10,50),AVCWT(50)
      REAL EOQ(50),ROP(50),STEP(3),PC(50),SEOQ(10,25,50)
      REAL SROP(10,25,50),SPC(10,50),TOTFIC(50),RPORD(20)
      REAL CMPRMR(50),MIC(50),SMIC(10,50),TOTRPC(50)
      REAL REPCAP(50),REPLD(50),TOTXC(50),ACQCST(50)
      REAL ACQLCS(20),FINACS(20),TOTLAQ(20),SAVAIL(10,10,50)
      REAL TOTSIC(10,50),TOTSXC(10,50),TOTSRP(10,50)
      REAL DISCST(50,10),SDISCS(10,50),AVAIL(50,10)
      REAL LCC(50),SLCC(10,50),SAVGAV(10,50),AVGAVL(50)
      REAL ENDPT(35,50),LSTAR(35,50),ASTAR(35,50)
      REAL ORD(10,50),LOP(10,50),SORDLC(10,10,50),
      REAL OBFN(5000),TEMP,SROPLC(10,10,50)
```

```
      REAL*8 ACCUM_A,ACCUM_B,USTIME

      CHARACTER*15 AQCOMP(50),COMPNM(50),COMP(50),LCOMP(50)
      CHARACTER*15 CMPNM(50),CM3,CM4
      CHARACTER*10 ACQLRU(50),LRU(50),LRUNUM(20),LRUSET(50,25)
      CHARACTER*10 NLRU(50)
      CHARACTER*4 ALOC(50,50),DEPOT(15,5),REPLOC(20,50)
      CHARACTER*4 REPSET(50,25),RPLOCN(60),STOPLC(50,25)
      CHARACTER*4 OPLOC(50), DEPLOC(10), REPLST(20,20)
      CHARACTER*4 LRUREP(5000,10),RPLC(50),RPCM(50)
      CHARACTER*3 COMPRQ(20,50),OPRQLS(50,25),BACK(50),
      CHARACTER*3 RQMT(25),RQ3,RQ4,LINK(35,10),STAGE(25)

      COMMON /RPBLK/ LRUREP,REPLST
      COMMON /PRDBLK/ LRUPRD,LPRDCL
      COMMON /OBJBLK/ OBFN

      DATA (RATIO(I),I=1,35)/35*1/
      DATA (CUMOP(I),I=1,50)/50*0.0/
      DATA (TOTCDM(I),I=1,50)/50*0.0/
      DATA ((TOTSDM(I,J),I=1,25),J=1,50)/1250*0.0/
      DATA ((CLRUDM(I,J),I=1,50),J=1,25)/1250*0.0/
      DATA (LT(I),I=1,50)/50*0.0/
      DATA ((SPC(I,J),I=1,10),J=1,50)/500*0.0/
      DATA (((CLDMLC(I,J,K),I=1,50),J=1,25),K=1,20)/25000*0.0/
      DATA (TOTFIC(I),I=1,50)/50*0.0/
      DATA (ACQCST(I),I=1,50)/50*0.0/
      DATA (ACQLCS(I),I=1,20)/20*0.0/
      DATA (FINACS(I),I=1,20)/20*0.0/
      RN=0


C ****
C Read in the model input file
C ****

      READ (10,101) NMCELL, NUMLOC, TMUNIT, TMIND, NUMDEP,
     +              NMCOMP, NMOPTN, GAMMA, NMRQMT, NUMLRU,
     +              NUMACQ, NMCMOP
101   FORMAT (2X,I5,I5,I5,I1,I5,I5,I5,I10,I5,I5,I5,I5)

      NMRPLC = NUMDEP + NUMLOC
      CELCNT = 0

      DO 10 I=1,NMCMOP
        READ (10,111) COMPNM(I), OPTN(NMOPTN+I)
111     FORMAT (2X, A15, I5)
10    CONTINUE
```

```
      DO 12 I=1,NMCOMP
        READ (10,112) CMPNM(I), LAMBDA(I), CSTPUR(I),
     +                NMPROD(I),NL(I)
112     FORMAT (2X, A15, F7.6, F8.2, I3, I3)
        CELCNT = CELCNT + NMPROD(I)
12    CONTINUE

      DO 20 I=1,NMCOMP
        READ (10,121) COMP(I), (PRDCEL(I,J), J=1,NMPROD(I))
121     FORMAT (2X, A15, 10I4)
20    CONTINUE

      DO 30 I=1,NMCELL
        READ(10,131)CELNUM(I),CSTLAB(I),CELCAP(I),CELOAD(I),
     +              CELLWT(I), CSTORD(I), HLDCST(I)
131     FORMAT (2X, I6, F8.2, F7.1, F7.1, F6.2, F6.2, F3.2)
30    CONTINUE

      DO 40 I=1,NMCOMP
        DO 45 J=1,NMPROD(I)
          READ(10,141) CSTSET(I,J),CYCLTM(I,J),SETPTM(I,J)
141       FORMAT(23X, F8.2, F8.2, F8.2)
45      CONTINUE
40    CONTINUE

      DO 50 I=1,NUMLRU
        READ(10,151)LRUNUM(I),CNDMRT(I),CSTDIS(I),NUMREP(I),
     +              NMCMRQ(I),NMCMP(I),LPROD(I)
151     FORMAT (2X, A10, F4.3, F8.2, I3, I4, I4, I4)
50    CONTINUE

      L=1
      DO 60 I=1,NUMLRU
        IF(NUMREP(I) .GT. 1) THEN
          REPLST(I,L)='OPLC'
          L=L+1
        ENDIF
        DO 65 J=1,NUMLOC+NUMREP(I)-1
          READ(10,161) REPLOC(I,J),LRUTST(I,J),LRURPL(I,J)
161       FORMAT (12X, A4, F4.1, F4.1)
          IF(J .GT. NUMLOC) THEN
            REPLST(I,L)=REPLOC(I,J)
            L=L+1
          ENDIF
65      CONTINUE
        L=1
60    CONTINUE
```

```
      DO 70 I=1,NUMLRU
        READ(10,171) (COMPRQ(I,J), J=1,NMCMRQ(I))
171     FORMAT (12X, 20A3)
70    CONTINUE

      DO 75 I=1,NUMLRU
         READ(10,176) (LPRDCL(I,J), J=1,LPROD(I))
176      FORMAT (12X,10I4)
75    CONTINUE

      DO 80 I=1,NMOPTN
        READ(10,181) OPTN(I), OPTIND(I), OPRQMT(I)
181     FORMAT (2X, I5, I1, I3)
80    CONTINUE

      DO 90 I=1,NMOPTN
        READ(10,191) (OPRQLS(I,J), J=1,OPRQMT(I))
191     FORMAT (7X, 15A3)
90    CONTINUE

      J=1
      L=1
      DO 100 I=1,NMRPLC
        READ(10,201) RPLOCN(I),CSTRPL(I),REPCAP(I),REPLD(I),
     +    OPTIME(I),CSTCHD(I),CSTLHD(I),DEPS(I),RPORD(I)
201     FORMAT (2X,A4,F8.2,F6.2,F6.2,F6.2,F4.3,F4.3,I3,F6.2)
        IF(I .LE. NUMLOC) THEN
          OPLOC(J)=RPLOCN(I)
          J=J+1
        ELSE
          DEPLOC(L)=RPLOCN(I)
          L=L+1
        ENDIF
100   CONTINUE

      DO 104 I=1,3
        READ (10,204) LOWAIT(I), UPWAIT(I)
204     FORMAT (5X, F6.2, F6.2)
104   CONTINUE

      DO 110 I=1,NMCMOP
         READ(10,211) LCOMP(I), LRU(I), COMTST(I), CMPRMR(I)
211      FORMAT (7X, A15, A10, F4.1, F5.2)
110   CONTINUE

      DO 120 I=1,NMRQMT
        READ(10,221) RQMT(I),OPTMRT(I),NUMOP(I),COMIND(I),
     +               NLINK(I)
221     FORMAT (2X, A3, F6.2, I4, I1,I3)
```

```
120    CONTINUE
       DO 130 I=1,NMRQMT
         READ(10,231) (RQMTOP(I,J), J=1,NUMOP(I))
231      FORMAT (5X, 15I4)
130    CONTINUE

       DO 135 I=1,NMRQMT
         READ(10,236) (LINK(I,J), J=1,NLINK(I))
236      FORMAT(5X, 15A3)
135    CONTINUE

c ****
c There could be multiple records for certain inputs, the next
c sections compute the number of records to be read
c ****

       LRUCNT = INT(NUMLRU/7)
       REMAIN = MOD(NUMLRU,7)
       L=0

       IF(LRUCNT .GT. 0) THEN
         DO 140 I=1,LRUCNT
           READ(10,241) (LRUSET(I,J), J=1,7)
241        FORMAT (2X, 7A10)
           DO 142 K=1,7
             L=L+1
             NLRU(L)=LRUSET(I,K)
142        CONTINUE
140      CONTINUE
       END IF

       IF(REMAIN .GT. 0) THEN
         LRUCNT = LRUCNT + 1
         L=L+1
         READ(10,245) (LRUSET(LRUCNT,J), J=1,REMAIN)
245      FORMAT (2X, 7A10)
         DO 147 K=1,REMAIN
           NLRU(L)=LRUSET(LRUCNT,K)
           L=L+1
147      CONTINUE
       END IF

       LOCCNT = INT(NUMLOC/15)
       RMNDR = MOD(NUMLOC,15)

       IF(LOCCNT .GT. 0)THEN
         DO 150 I=1,LOCCNT
           READ(10,251) (STOPLC(I,J), J=1,15)
251        FORMAT (2X, 15A4)
150      CONTINUE
```

```
      END IF
      IF(RMNDR .GT. 0) THEN
        LOCCNT = LOCCNT + 1
        READ(10,255) (STOPLC(LOCCNT,J), J=1,RMNDR)
255     FORMAT (2X, 15A4)
      END IF

      REPCNT = INT(NMRPLC/15)
      REMNDR = MOD(NMRPLC,15)

      IF(REPCNT .GT. 0) THEN
        DO 160 I=1,REPCNT
          READ (10,261) (REPSET(I,J), J=1,15)
261       FORMAT (2X, 15A4)
160     CONTINUE
      END IF

      IF (REMNDR .GT. 0) THEN
        REPCNT = REPCNT + 1
        READ(10,265) (REPSET(REPCNT,J), J=1,REMNDR)
265     FORMAT (2X, 15A4)
      END IF

      DO 170 I=1,NUMLOC
        DEPCNT = DEPS(I) + 1
        READ (10,271) (TRNCST(I,J), TRANTM(I,J), DEPOT(I,J),
     +                 J=1,DEPCNT)
271     FORMAT (6X, 4(F6.2, F3.1, A4))
170   CONTINUE

      DO 180 I=1,NUMACQ
        READ (10,281) TIME(I), ACQ(I), ACQLOC(I)
281     FORMAT (2X, I5, I5, I5)
180   CONTINUE

      K=0
      DO 190 I=1,NUMACQ
        ACQCNT=INT(ACQLOC(I)/7)
        ACREM=MOD(ACQLOC(I),7)
        IF (ACQCNT .GT. 0) THEN
          DO 195 J = 1,ACQCNT
            K = K + 1
            READ(10,291) (NACQ(K,L), ALOC(K,L), L=1,7)
291         FORMAT (7X, 7(I5, A4))
195       CONTINUE
        END IF
```

```
            IF(ACREM .GT. 0) THEN
              K = K + 1
              READ (10, 295) (NACQ(K,L), ALOC(K,L), L=1,ACREM)
295           FORMAT (7X, 7(I5, A4))
            END IF
190   CONTINUE

      L = 0
      DO 200 I=1,NMCOMP
        ACNT = INT(NUMACQ/5)
        AREM = MOD(NUMACQ,5)
        IF (ACNT .GT. 0) THEN
          DO 205 J=1,ACNT
            L = L + 1
            READ(10,301) AQCOMP(L), (ACQN(L,K), CMACST(L,K),
K=1,5)
301         FORMAT (2X, A15, 5(I3, F8.2))
205       CONTINUE
        END IF

            IF(AREM .GT. 0) THEN
              L = L + 1
              READ(10,305) AQCOMP(L), (ACQN(L,K), CMACST(L,K),
     +                       K = 1,AREM)
305           FORMAT (2X, A15, 5(I3, F8.2))
            END IF
200   CONTINUE

      M = 0
      DO 210 I=1,NUMLRU
        ACQCT = INT(NUMACQ/3)
        ACRMND = MOD(NUMACQ,3)
        IF (ACQCT .GT. 0) THEN
          DO 215 J=1,ACQCT
            M = M + 1
            READ(10,311) ACQLRU(M), (LACQN(M,K), LRUCST(M,K),
     +                     FINCST(M,K), K=1,3)
311         FORMAT (2X, A10, 6F8.2)
215       CONTINUE
        END IF

            IF(ACRMND .GT. 0) THEN
              M = M + 1
            READ(10,315) ACQLRU(M), (LACQN(M,K), LRUCST(M,K),
     +                     FINCST(M,K), K=1,ACRMND)
315         FORMAT (2X, A10, 3(I3,F8.2,F8.2))
            END IF
210   CONTINUE
```

```
c ****
c * These subroutines compute the intermediate performance
c * measures to build LCC and system availability
c ****

      CALL BLDBK(NMRQMT,RQMT,NUMOP,RQMTOP,OPTN,BACK)
      CALL BLDOBK(NMOPTN,RQMT,OPRQMT,OPRQLS,OPTN,OBACK)
      CALL BLDDH(NMRQMT,COMIND,DH)
      CALL BLDTH(NMOPTN,OPTIND,OPRQMT,OPRQLS,RQMT,TH)
      CALL BLDSTG(NMRQMT,BACK,OBACK,RQMT,OPRQLS,RQMTOP,
     +       RATE,DH,OPRQMT,NUMOP,COMIND,OPTIND,OPTN,
     +       STAGE,TH,RT,LAMBDA,NMOPTN,H,COMP,COMPNM,
     +       CNT,ENM,NR)
      CALL OPRATE(DMNDRT,RATE,RATIO,ORATIO,OPTMRT,
     +       RQMT,BACK,LAMBDA,NMRQMT,COMIND,OBACK,NUMOP,
     +       RQMTOP,NMCMOP,OPTN,FAILRT,COMPNM,COMP)
      CALLOPERTM(NUMACQ,ACQLOC,NACQ,ALOC,RPLOCN,TMUNIT,TIME,
     +       OPTIME,CUMOP)
      CALL STNDMD(NMRQMT,OBACK,OPTN,COMIND,OPTIND,OPRQMT,
     +       RT,SDMND,H,CNT)
      CALL DEMAND(NUMLOC,NMCMOP,CUMOP,DMNDRT,ORATIO,CDMDLC,
     +       SDMDLC,RATIO,RQMT,OPRQLS,COMIND,OPTIND,SDMND,
     +       NMOPTN,CNT,TOTCDM,TOTSDM)
      CALL LRUROL(CLRUDM,LCOMP,NMCMOP,LRU,NLRU,RQMT,BACK,
     +       OBACK,TOTCDM,COMP,NMOPTN,OPTN,NUMLOC,CUMOP,
     +       CLDMLC)
      CALL LSTREP(NUMLRU,NUMREP,RCNT)
      CALL LSTPRD(NUMLRU,LPROD,PCNT)

c ****
c * The next set of loops roll up the acquisition cost for the
c * LRU's and the component's.  This can be done outside the
c * D.P. loop.
c ****

      DO 600 I=1,NUMACQ
        DO 605 J=1,NMCOMP
          ACQCST(J)=CMACST(J,I)*ACQ(I)+ACQCST(J)
605     CONTINUE
600   CONTINUE

      DO 610 I=1,NUMACQ
        DO 615 J=1,NUMLRU
          ACQLCS(J)=LRUCST(J,I)*ACQ(I)+ACQLCS(J)
          FINACS(J)=FINCST(J,I)*ACQ(I)+FINACS(J)
615     CONTINUE
610   CONTINUE
```

```
      DO 620 I=1,NUMLRU
        TOTLAQ(I)=ACQLCS(I)+FINACS(I)
620   CONTINUE

c ****
c * Subroutine DISPOS performs the disposal cost computation
c ****

      CALL DISPOS(NMCOMP,NUMLRU,TMUNIT,CNDMRT,CSTDIS,
     +            NMOPTN,OPTIND,CNT,DISCST,ENM,LRU,NLRU,
     +            SDISCS,CLRUDM,TOTSDM)

c ****
c * This is the outer problem of the D.P.  It passes the
c * average wait time to the inner problem
c ****

        STEP=(UPWAIT-LOWAIT)/10

      DO 907 X101=LOWAIT,UPWAIT,STEP
c ****
c * This routine takes the repair set, input by LRU and
c * builds the list of repair locations for the middle
c * problem
c ****
      DO 910 I110=1,RCNT
       DO 912 I112=1,NUMLRU
        RPLC(I112)=LRUREP(I110,I112)
        DO 924 I124=1,NMCMRQ(I112)
         K2=0
         RQ4=COMPRQ(I112,I124)
         CALL RQR(K2,RQ4,RQMT)
         DO 926 J126=1,NUMOP(K2)
          OP1=RQMTOP(K2,J126)
          C2=0
          CM4=COMPNM(OP1-NMOPTN)
          CALL CMPRTN(C2,CM4,COMP)
          RPCM(C2)=RPLC(I112)
926      CONTINUE
924     CONTINUE
912    CONTINUE
c ****
c * These routines build the repair cost
c ****
       CALL REPCST(CSTRPL,COMTST,LRUTST,LRURPL,CMPRMR,RPLC,
     +     LRU,NLRU,REPLOC,COMP,NUMLOC,NMCMOP,NMCOMP,TOTRPC,
     +      REPLD,TMUNIT,LCOMP,NUMLRU,CDMDLC)
       CALL STNREP(CSTRPL,COMTST,LRUTST,LRURPL,CMPRMR,RPLC,
     +       LRU,NLRU,REPLOC,NUMLOC,OPTIND,ENM,NMOPTN,
     +       TOTSRP,REPLD,TMUNIT,CNT,OPRQMT,SDMDLC)
```

```
         DO 930 K5=1,NMCOMP
          AVCWT(K5)=X101/250
930      CONTINUE
c ****
c * This routine takes the production location set for each
c * LRU and builds the production list for the middle problem
c ****
        DO 914 I114=1,PCNT
         DO 916 I116=1,NUMLRU
          PRDN(I116)=LRUPRD(I114,I116)
          DO 920 I120=1,NMCMRQ(I116)
           K1=0
           RQ3=COMPRQ(I116,I120)
           CALL RQR(K1,RQ3,RQMT)
           DO 922 J122=1,NUMOP(K1)
            OP=RQMTOP(K1,J122)
            C1=0
            CM3=COMPNM(OP-NMOPTN)
            CALL CMPRTN(C1,CM3,COMP)
            PDCM(C1)=PRDN(I116)
922        CONTINUE
920       CONTINUE
916      CONTINUE
c ****
c * These routines compute the production batch size and
c * reorder point for the components at the manufacturer's
c ****
         CALL LEADTM(NMCOMP,NUMLRU,CLRUDM,CSTORD,HLDCST,CSTPUR,
     +           PDCM,SETPTM,CYCLTM,CELOAD,LT,PRDCEL,TMUNIT,
     +           CELCAP,AVCWT,EOQ,ROP,CSTSET,CSTLAB,PC,MIC)
         CALL SLDTM(CNT,TOTSDM,PDCM,CSTORD,HLDCST,CSTPUR,
     +         SETPTM,CYCLTM,CELOAD,SLT,PRDCEL,TMUNIT,CELCAP,
     +           NMOPTN,OPTIND,COMP,COMPNM,OPRQMT,ENM,SEOQ,
     +           SROP,AVCWT,SPC,CSTSET,CSTLAB,SMIC)
c ****
c * These routines compute the field inventory cost by
c * operating location and repair location.  They also compute
c * the transportation costs.
c ****
         CALL FLDINV(NUMLOC,CLDMLC,NMCOMP,NUMLRU,RPCM,DEPOT,
     +           TRANTM,AVCWT,CSTCHD,CSTPUR,RPORD,TOTFIC,LT,
     +          TMUNIT,TRNCST,TOTXC,DMNDRT,COMPNM,COMP,AVAIL,
     +           NMCMOP,ORD,LOP)
         CALL STNINV(NUMLOC,SDMDLC,NMOPTN,RPLC,OPTIND,
     +        DEPOT,TRANTM,AVCWT,CSTCHD,CSTPUR,RPORD,TOTSIC,
     +        SLT,TMUNIT,TRNCST,TOTSXC,COMP,LRU,COMPNM,NLRU,
     +        CNT,OPRQMT,ENM,RT,SAVAIL,SORDLC,SROPLC)
```

```
c ****
c * This routine checks the configuration options to see if
c * any are dominated according to their cost and availability
c ****
         CALL DMNATE(NUMLOC,NMOPTN,COMIND,OPTIND,CNT,OPRQLS,
      +        OPRQMT,COMPNM,COMP,NUMOP,RQMT,DSFLG,DFLG,
      +       AVAIL,SAVAIL,ACQCST,TOTXC,TOTFIC,MIC,PC,TOTSXC,
      +      TOTSIC,SMIC,SPC,TOTRPC,TOTSRP,DISCST,SDISCS,LCC,
      +        SLCC,RQMTOP,NMCOMP,NUMLRU,ENM,NMRQMT,
      +        SAVGAV,AVGAVL)
         CALL DECRUL(STAGE,NR,RQMT,COMIND,OPTIND,CNT,OBACK,
      +       DSFLG,DFLG,SLCC,SAVGAV,LCC,AVGAVL,GAMMA,
      +       NMOPTN,NUMOP,RQMTOP,COMPNM,COMP,OPRQLS,LINK,
      +       STGCNT,ENDPT,LSTAR,ASTAR,CHOICE)
         CALLPLAYBK(NR,STAGE,RQMT,COMIND,OPTIND,STGCNT,ENDPT,
      +       CHOICE,OPRQLS,LSTAR,ASTAR,OBACK,AVGAVL,SAVGAV,
      +       LINK,OPRQMT,NMOPTN,ENM,GAMMA,RPLC,PRDN,NUMLRU,
      +       EOQ,ROP,SEOQ,SROP,ORD,LOP,SORDLC,SROPLC,RN,
      +       COMPNM,COMP)
914     CONTINUE
910    CONTINUE
907    CONTINUE

c ****
c * Compares the objective values for each run of the inner
c * D.P. to determine which run is the lowest
c ****
       TEMP=1000000.
       DO 1100 I=1,RN
         IF(OBFN(I) .LT. TEMP) THEN
           TEMP=OBFN(I)
           RUN=I
         ENDIF
1100   CONTINUE
       WRITE(20,*) 'MIN OBJ IS ',RUN
       WRITE(20,*) 'NUMBER OF RUNS IS ',RN
       END

c ****
c * Finds the preceeding requirement in the product structure
c * for each option
c ****

       SUBROUTINE BLDBK(NMRQMT,RQMT,NUMOP,RQMTOP,OPTN,BACK)
       CHARACTER*3 BACK(50),RQMT(25)
       INTEGER NUMOP(50),NMRQMT,RQMTOP(30,5),OPTN(50)
       INTEGER OP
       K=0
```

```
      DO 300 I=1,NMRQMT
          DO 307 J=1,NUMOP(I)
            OP=RQMTOP(I,J)
2000        K=K+1
            IF (OPTN(K) .NE. OP) GOTO 2000
            BACK(K) = RQMT(I)
            K=0
307       CONTINUE
300   CONTINUE
      RETURN
      END

c ****
c * Finds the preceeding option for each requirement in the
c * product structure.  Helps to travel through the network.
c ****

      SUBROUTINEBLDOBK(NMOPTN,RQMT,OPRQMT,OPRQLS,OPTN,OBACK)
      CHARACTER*3 OPRQLS(50,25),RQMT(25),RQ
      INTEGER OPRQMT(50),NMOPTN,OPTN(50),OBACK(50)
      OBACK(1)=0
      K=0
      DO 310 I=1,NMOPTN
          DO 318 J=1,OPRQMT(I)
            RQ=OPRQLS(I,J)
2005        K=K+1
            IF(RQMT(K) .NE. RQ) GOTO 2005
            OBACK(K) = OPTN(I)
            K=0
318       CONTINUE
310   CONTINUE
      RETURN
      END

c ****
c Assists in building the 'h' value for MTBF recursion
c ****

      SUBROUTINE BLDDH(NMRQMT,COMIND,DH)
      INTEGER NMRQMT,COMIND(25)
      REAL DH(25)
      DO 320 I=1,NMRQMT
        IF(COMIND(I) .EQ. 0) THEN
          DH(I) = 0.0
        ELSE
          DH(I) = 1.0
        ENDIF
320   CONTINUE
      END
```

```
c ****
c Assists in building the 'h' value for MTBF recursion.  Also
c decrements the failure rate for sub-assemblies in parallel
c configuration.
c ****

      SUBROUTINE BLDTH(NMOPTN,OPTIND,OPRQMT,OPRQLS,RQMT,TH)
      INTEGER NMOPTN,OPTIND(50),OPRQMT(50)
      CHARACTER*3 OPRQLS(50,25),RQ,RQMT(25)

      REAL TH(25)
      P=0
      R=0
      TH(1) = 1.0
      DO 330 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 2) THEN
          RQ=OPRQLS(I,1)
          DO 335 J=1,OPRQMT(I)
            R = R + 1
            P = P + 1/R
335       CONTINUE
        K=0
        CALL RQR(K,RQ,RQMT)
        TH(K) = 1/P
        P=0
        R=0
        ENDIF
        IF((OPTIND(I) .EQ. 1) .OR. (OPTIND(I) .EQ. 3)) THEN
          DO 338 J=1,OPRQMT(I)
            K=0
            RQ = OPRQLS(I,J)
            CALL RQR(K,RQ,RQMT)
            TH(K)=1.0
338       CONTINUE
        ENDIF
330   CONTINUE
      END

c ****
c Builds the stages for the inner product d.p.
c ****

      SUBROUTINEBLDSTG(NMRQMT,BACK,OBACK,RQMT,OPRQLS,RQMTOP,
     +     RATE,DH,OPRQMT,NUMOP,COMIND,OPTIND,OPTN,STAGE,
     +     TH,RT,LAMBDA,NMOPTN,H,COMP,COMPNM,CNT,ENM,NR)

      INTEGER NMRQMT,OBACK(50),RQMTOP(30,5),O(25),R(25)
      INTEGER CNT(25),OPNUMOP(50),COMIND(25),OPRQMT(50)
      INTEGER ENM(5,10,25),OPTIND(50),OPTN(50)
```

```
      CHARACTER*15 COMP(50),COMPNM(50)
      CHARACTER*3 BACK(50),OPRQLS(50,25),STAGE(25),RQ,RQMT(25)
      REAL CUM(25),RATE(50),TH(25),H(50),DH(25),RT(10,50)
      REAL LAMBDA(50)

      DATA (CUM(K),K=1,25)/25*1/
      I=0
      RQ=RQMT(1)
      IFLG=1
      JFLG=1
      NR=NMRQMT-1

c ****
c * Flag is set depending on type of node the requirement is
c ****
2007  IF(JFLG .EQ. 1) THEN
      K=0
      CALL RQR(K,RQ,RQMT)

c ****
c * O(K) represents the number of options per requirement.
c * It keeps an index of which option has been selected.
c ***
      O(K)=NUMOP(K)
      END IF

c ****
c * If the requirement is a component requirement, then we
c * have a stage.
c ****

      IF(COMIND(K) .EQ. 1) THEN
        I = I + 1
        STAGE(I) = RQ
c ****
c * R(N) represents the number of requirements per option.
c * It keeps an index of which requirement has been chosen.
c ***
        R(N) = R(N) - 1
        IF(R(N) .GT. 0) THEN
          RQ = OPRQLS(N,R(N))
        ELSE
          RQ=BACK(N)
          K=0
          CALL RQR(K,RQ,RQMT)
          O(K) = O(K) - 1
          JFLG=0
        ENDIF
```

```
c ****
c * When O(K) is 0, we have traveled down each path beneath a
c * particular requirement
c ****
392      IF(O(K) .EQ. 0) THEN
           I=I+1
           STAGE(I) = RQ
           OP=OBACK(K)
           JFLG=1
           N=0
           CALL OPN(N,OP,OPTN)
           R(N) = R(N) - 1
c ****
c * When R(N) is 0, we have traveled down each path beneath a
c * particular option.
c ****
           IF(R(N) .GT. 0) THEN
             RQ=OPRQLS(N,R(N))
           ELSE
             IF(N .EQ. 1) THEN
               GOTO 391
             ELSE
               RQ=BACK(N)
               K=0
               CALL RQR(K,RQ,RQMT)
               O(K) = O(K) - 1
               JFLG=0
               IF(O(K).EQ.0) GOTO 392
             ENDIF
           ENDIF
         ENDIF
       IFLG = 0
       ENDIF

       IF(IFLG .EQ. 1) THEN
         JFLG=1
         OP = RQMTOP(K,O(K))
         N=0
         CALL OPN(N,OP,OPTN)
         R(N) = OPRQMT(N)

         IF(OPTIND(N) .EQ. 3) THEN
c ****
c * Enumerates the options for standby assemblies
c ****
           CALL STNDBY(K,N,OPRQLS,OPRQMT,NUMOP,LAMBDA,RT,
     +          RQMTOP,RQ,R,COMIND,NMOPTN,RQMT,CNT,COMP,
     +          COMPNM,ENM)
           NR=NR-1
         ENDIF
```

```
          IF(OPTIND(N) .EQ. 2) THEN
            R(N)=1
            RQ=OPRQLS(N,R(N))
          ENDIF

          IF(OPTIND(N) .EQ. 1) THEN
            RQ=OPRQLS(N,R(N))
          ENDIF
        ENDIF
        IFLG=1
        IF(I .NE. NR) GOTO 2007

c ****
c * Runs through the requirements at the end of building the
c * stages to build the 'h' value for MTBF recursion
c ****

391     DO 340 I=NMRQMT,2,-1
          IF(COMIND(I) .EQ. 1) THEN
            RQ=RQMT(I)
2009        K=0
            CALL RQR(K,RQ,RQMT)
            OP=OBACK(K)
            N=0
            CALL OPN(N,OP,OPTN)
            RQ=BACK(N)
            CUM(I)=CUM(I)*TH(K)
            IF(N .NE. 1) GOTO 2009
          ENDIF
340     CONTINUE

        DO 350 I=1,NMRQMT
          H(I)=CUM(I)*DH(I)
350     CONTINUE

c ****
c Gives the 'h' value associated with each requirement to the
c component options beneath it in the product structure
c ****
        K=0
        DO 360 I=1,NMRQMT
          IF(COMIND(I) .EQ. 1) THEN
            DO 365 J=1,NUMOP(I)
              OP=RQMTOP(I,J)
              K=K+1
              RATE(K)=H(I)
365         CONTINUE
          ENDIF
360     CONTINUE
        END
```

```
c ****
c * Finds an index value for each requirement since they are
c * defined as character
c ****
      SUBROUTINE RQR(K,RQ,RQMT)
      CHARACTER*3 RQ,RQMT(25)
2011  K=K+1
      IF(RQMT(K).NE.RQ) GOTO 2011
      END
c ****
c * Finds an index value for each option since they are in
c * no particular order
c ****
      SUBROUTINE OPN(N,OP,OPTN)
      INTEGER OPTN(50), OP
2013  N=N+1
      IF(OPTN(N).NE.OP) GOTO 2013
      END

c ****
c * This subroutine is for stand-by sub-assy's to combine the
c * failure rates and decrement the stage number
c ****

      SUBROUTINE STNDBY(K,N,OPRQLS,OPRQMT,NUMOP,LAMBDA,
     +  RT,RQMTOP,RQ,R,COMIND,NMOPTN,RQMT,CNT,COMP,
     +  COMPNM,ENM)

      INTEGER N,K,OPRQMT(50),NUMOP(50),RQMTOP(30,5)
      INTEGER R(25),COMIND(25),NMOPTN,CNT(25),ENM(5,10,25)
      REAL LAMBDA(50),RT(10,50)
      CHARACTER*3 OPRQLS(50,25),RQ,RQMT(25)
      CHARACTER*15 COMP(50),COMPNM(50)
      RQ=OPRQLS(N,1)
      K=0
      CALL RQR(K,RQ,RQMT)
      IF (COMIND(K) .EQ. 1) THEN
        CALL ENUM(N,OPRQMT,NUMOP,LAMBDA,RT,RQMTOP,NMOPTN,
     +       RQMT,OPRQLS,COMP,COMPNM,CNT,ENM)
        R(N)=1
      ENDIF
      RQ=OPRQLS(N,R(N))
      END

c ****
c * This routine enumerates all of the options of stand-by
c * sub-assy's which are also component requirements to
c * combine the MTBF's
c ****
```

```
      SUBROUTINEENUM(N,OPRQMT,NUMOP,LAMBDA,RT,RQMTOP,NMOPTN,
     +                    RQMT,OPRQLS,COMP,COMPNM,CNT,ENM)
      INTEGER N,OPRQMT(50),NUMOP(50),RQMTOP(30,5),NMOPTN
      INTEGER OP,ENM(5,10,25),FLG,SOP(25),C1,CNT(25)
      REAL LAMBDA(50), RT(10,50), MTBF
      CHARACTER*15 CM,COMP(50),COMPNM(50)
      CHARACTER*3 OPRQLS(50,25),RQ,RQMT(25)
c ****
c * The value of CNT is the number of options for the standby
c * assembly after the options have been enumerated
c ****
      CNT(N)=1
      DO 430 I=1,OPRQMT(N)
        RQ=OPRQLS(N,I)
        K=0
        CALL RQR(K,RQ,RQMT)
        CNT(N)=CNT(N)*NUMOP(K)
430   CONTINUE

          DO 443 I2=1,OPRQMT(N)
            RQ=OPRQLS(N,I2)
            K5=0
            CALL RQR(K5,RQ,RQMT)
            SOP(I2)=NUMOP(K5)
443       CONTINUE
          I=1
          J=OPRQMT(N)
          K=J-1
          FLG=0
2015      IF(FLG .EQ. 0) THEN
c ****
c * Starts with the last option of the first requirement in
c * the assembly and matches it with each option of the
c * rest of the requirements
c ****
            DO 444 J5=1,NUMOP(J)
2017          RQ=OPRQLS(N,J)
              K5=0
              CALL RQR(K5,RQ,RQMT)
              ENM(N,I,J)=RQMTOP(K5,SOP(J))
              J=J-1
              IF(J .NE. 0) GOTO 2017
              I=I+1
              J=OPRQMT(N)
              SOP(J)=SOP(J)-1
444         CONTINUE
          ENDIF
          SOP(J)=NUMOP(J)
          SOP(K)=SOP(K)-1
```

```
c ****
c * Then moves to the next option of the first requirement
c * and continues the process
c ****
          IF((FLG .EQ. 1) .AND. (SOP(K) .NE. 0)) THEN
            IF(K .EQ. 1) THEN
              K=OPRQMT(N)-1
            ELSE
              K=K+1
            ENDIF
          ENDIF
          FLG=0
          IF(SOP(K) .EQ. 0) THEN
            SOP(K)=NUMOP(K)
            FLG=0
            K=K-1
          ENDIF
          IF(K .NE. 0) GOTO 2015
c ****
c * Builds the demand rate for each 'new' enumerated option
c * in the standby assembly
c ****
      MTBF=0
      DO 410 I=1,CNT(N)
        DO 415 J=OPRQMT(N),1,-1
          OP=ENM(N,I,J)
          CM=COMPNM(OP-NMOPTN)
          C1=0
          CALL CMPRTN(C1,CM,COMP)
          MTBF=MTBF+(1/LAMBDA(C1))
415     CONTINUE
        RT(N,I)=1/MTBF
        MTBF=0
410   CONTINUE
      END

c ****
c * Rolls up the operating time ratio for each requirement
c * in the product structure
c ****
      SUBROUTINEOPRATE(DMNDRT,RATE,RATIO,ORATIO,OPTMRT,RQMT,
     +    BACK,LAMBDA,NMRQMT,COMIND,OBACK,NUMOP,RQMTOP,
     +      NMCMOP,OPTN,FAILRT,COMPNM,COMP)

      REAL DMNDRT(50), RATE(50), RATIO(35), ORATIO(35)
      REAL LAMBDA(50), FAILRT(50),OPTMRT(35)
      CHARACTER*3 BACK(50),RQ,RQMT(25)
      CHARACTER*15 COMPNM(50), COMP(50), CM
      INTEGER NMRQMT,COMIND(25),OBACK(50),NUMOP(50)
      INTEGER NMCMOP,OPTN(50),C1,OP,RQMTOP(30,5)
```

```
c ****
c * Takes the failure rate, which is input by component and
c * gives it to each component option
c ****
      DO 375 I=1,NMCMOP
        CM=COMPNM(I)
        C1=0
        CALL CMPRTN(C1,CM,COMP)
        FAILRT(I)=LAMBDA(C1)
375   CONTINUE
c ****
c * Computes the demand rate, which is the failure rate
c * augmented by the value of 'h'.
c ****

      DO 370 I=1,NMCMOP
        DMNDRT(I) = RATE(I) * FAILRT(I)
370   CONTINUE

c ****
c * Proceeds through the product structure from the bottom
c * up until it reaches option 1
c ****

      DO 380 I=NMRQMT,2,-1
        IF (COMIND(I) .EQ. 1) THEN
          RQ=RQMT(I)
2019      K=0
          CALL RQR(K,RQ,RQMT)
          OP=OBACK(K)
          N=0
          CALL OPN(N,OP,OPTN)
          RQ=BACK(N)
          RATIO(I)=RATIO(I)*OPTMRT(K)
          IF(N .NE. 1) GOTO 2019
        ENDIF
380   CONTINUE

      K=0
      DO 390 I=1,NMRQMT
        IF (COMIND(I) .EQ. 1) THEN
          DO 395 J=1,NUMOP(I)
            OP=RQMTOP(I,J)
            K=K+1
            ORATIO(K)=RATIO(I)
395       CONTINUE
        ENDIF
390   CONTINUE
      END
```

```
c ****
c * This subroutine computes the operating time for each
c * location, subject to the number of products purchased
c * for each location during acquisition
c ****
      SUBROUTINE OPERTM(NUMACQ,ACQLOC,NACQ,ALOC,RPLOCN,
     +     TMUNIT,TIME,OPTIME,CUMOP)
      REAL OPTIME(60),CUMOP(50),OPTM(50)
      INTEGER NUMACQ,ACQLOC(50),NACQ(25,50),TMUNIT,TIME(25),AC
      CHARACTER*4 ALOC(50,50),RPLOCN(60),LC
c ****
c * Determines the number of acquisition records
c ****
      DO 317 I=1,NUMACQ
        ACQCNT=INT(ACQLOC(I)/7)
        ACREM=MOD(ACQLOC(I),7)
        IF(ACQCNT .GT. 0) THEN
          DO 327 J=1,ACQCNT
            DO 337 K=1,7
              NMACQ=NACQ(J,K)
              LC=ALOC(J,K)
              L=0
              CALL LOC(L,LC,RPLOCN)
              OPTM(L)=(TMUNIT - TIME(I) + 1)*NMACQ*OPTIME(L)
              CUMOP(L) = CUMOP(L) + OPTM(L)
337         CONTINUE
327       CONTINUE
        ENDIF
        AC=ACQCNT+1
        IF (ACREM .GT. 0) THEN
          DO 347 J=1,ACREM
            NMACQ=NACQ(AC,J)
            LC=ALOC(AC,J)
            L=0
            CALL LOC(L,LC,RPLOCN)
            OPTM(L)=(TMUNIT - TIME(I) + 1)*NMACQ*OPTIME(L)
            CUMOP(L) = CUMOP(L) + OPTM(L)
347       CONTINUE
        ENDIF
317   CONTINUE
      END
c ****
c * Finds an index value for each location, since they are
c * defined as character values
c ****
      SUBROUTINE LOC(L,LC,RPLOCN)
      CHARACTER*4 LC,RPLOCN(60)
2021  L=L+1
      IF(RPLOCN(L) .NE. LC) GOTO 2021
      END
```

```
c ****
c * This routine takes the combined failure rates for the
c * standby assy's and multiplies by 'h' to get demand rate
c ****

      SUBROUTINE STNDMD(NMRQMT,OBACK,OPTN,COMIND,OPTIND,
     +     OPRQMT,RT,SDMND,H,CNT)

      INTEGERNMRQMT,OBACK(50),OPTN(50),COMIND(25),OPTIND(50)
      INTEGER OPRQMT(50),CNT(25),OP
      REAL RT(10,50),SDMND(25,50),H(50)

      STEP=-1
      DO 357 I=NMRQMT,2,STEP
        STEP=-1
        N2=0
        OP=OBACK(I)
        N1=0
        CALL OPN(N1,OP,OPTN)
        IF ((COMIND(I) .EQ. 1) .AND. (OPTIND(N1) .EQ. 3)) THEN
          STEP=-OPRQMT(N1)
          DO 367 J=1,CNT(N1)
            SDMND(N1,J)=RT(N1,J)*H(I)
367       CONTINUE
        ENDIF
357   CONTINUE
      END

c ****
c * This routine computes the expected number of demands for
c * each component
c ****
      SUBROUTINE DEMAND(NUMLOC,NMCMOP,CUMOP,DMNDRT,ORATIO,
     +       CDMDLC,SDMDLC,RATIO,RQMT,OPRQLS,COMIND,OPTIND,
     +       SDMND,NMOPTN,CNT,TOTCDM,TOTSDM)
      INTEGER NUMLOC,NMCMOP,COMIND(25),OPTIND(50),NMOPTN
      INTEGER CNT(25)
      REAL CUMOP(50),DMNDRT(50),ORATIO(35),CDMDLC(20,50)
      REAL SDMDLC(20,20,50),SDMND(25,50),RATIO(35)
      REAL TOTCDM(50),TOTSDM(25,50)
      CHARACTER*3 OPRQLS(50,25),RQMT(25),RQ4,RQ5
c ****
c * This rolls up component option demand by location
c ****
      DO 312 I=1,NUMLOC
        DO 322 J=1,NMCMOP
          CDMDLC(I,J)=CUMOP(I)*DMNDRT(J)*ORATIO(J)
322     CONTINUE
312   CONTINUE
```

```
c ****
c * This computes standby demands by location
c ****

      DO 332 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 3) THEN
          RQ4=OPRQLS(I,1)
          K=0
          CALL RQR(K,RQ4,RQMT)
          IF (COMIND(K) .EQ. 1) THEN
            DO 342 J=1,NUMLOC
              DO 352 L=1,CNT(I)
                SDMDLC(I,L,J)=RATIO(K)*SDMND(I,L)*CUMOP(J)
352           CONTINUE
342         CONTINUE
          ENDIF
        ENDIF
332   CONTINUE

c ****
c * This is total component demand across all locations
c ****

      DO 362, I=1,NMCMOP
        DO 372 J=1,NUMLOC
          TOTCDM(I)=TOTCDM(I)+CDMDLC(J,I)
372     CONTINUE
362   CONTINUE

c ****
c * This is total standby demand across all locations
c ****

      DO 382 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 3) THEN
          RQ5=OPRQLS(I,1)
          K=0
          CALL RQR(K,RQ5,RQMT)
          IF(COMIND(K) .EQ. 1) THEN
            DO 392 L=1,CNT(I)
              DO 302 J=1,NUMLOC
                TOTSDM(I,L) = TOTSDM(I,L) + SDMDLC(I,L,J)
302           CONTINUE
392         CONTINUE
          ENDIF
        ENDIF
382   CONTINUE
      END
```

```
c ****
c * This routine computes an index value for each component
c ****
      SUBROUTINE CMPRTN(C1,CM,COMP)
      INTEGER C1
      CHARACTER*15 CM,COMP(50)
2023  C1=C1+1
      IF(CM .NE. COMP(C1)) GOTO 2023
      END
c ****
c * This routine rolls up the component demand for each LRU
c * if there are multiple occurances of the component in
c * the LRU.
c ****

      SUBROUTINE LRUROL(CLRUDM,LCOMP,NMCMOP,LRU,NLRU,RQMT,
     +       BACK,OBACK,TOTCDM,COMP,NMOPTN,OPTN,NUMLOC,
     +       CUMOP,CLDMLC)
      INTEGER OP,OP1,C1,OBACK(50),NMCMOP,LFLG(50),C2
      INTEGER NMOPTN, OPTN(50), IFLG(50), NUMLOC
      REAL TOTCDM(50),CLRUDM(50,25),CLDMLC(50,25,20),CUMOP(50)
      REAL PCTOP(50)
      CHARACTER*10 LRU(50),NLRU(50),LR
      CHARACTER*15 COMP(50),LCOMP(50),CM
      CHARACTER*3 RQMT(25),BACK(50),RQ,RQ1

      DATA (LFLG(I), I=1,50)/50*0/
      DATA (IFLG(I), I=1,50)/50*0/

      TOTOP=0
      DO 423 I=1,NUMLOC
        TOTOP=TOTOP+CUMOP(I)
423   CONTINUE

      DO 425 I=1,NUMLOC
        PCTOP(I)=CUMOP(I)/TOTOP
425   CONTINUE

      DO 420 I=1,NMCMOP-1
        CM=LCOMP(I)
        C1=0
        CALL CMPRTN(C1,CM,COMP)
        IF (LFLG(I) .EQ. 0) THEN
          DO 422 J=I+1,NMCMOP
            CM=LCOMP(J)
            C2=0
            CALL CMPRTN(C2,CM,COMP)
            LR=LRU(I)
            L1=0
            CALL LRURTN(L1,LR,NLRU)
```

```
                IF (IFLG(I) .EQ. 0) THEN
                CLRUDM(C1,L1)=TOTCDM(I)
                DO 426 K3=1,NUMLOC
                   CLDMLC(C1,L1,K3) = CLRUDM(C1,L1)*PCTOP(K3)
426             CONTINUE
                END IF
                IF ((LCOMP(I).EQ.LCOMP(J)) .AND.
     +               LRU(I).EQ.LRU(J))  THEN
                  OP=OPTN(C1+NMOPTN)
                  N=0
                  CALL OPN(N,OP,OPTN)
                  OP1=OPTN(C2+NMOPTN)
                  N1=0
                  CALL OPN(N1,OP1,OPTN)
2025              RQ=BACK(N)
                  RQ1=BACK(N1)
                  K=0
                  K1=0
                  CALL RQR(K,RQ,RQMT)
                  CALL RQR(K1,RQ1,RQMT)
                  OP=OBACK(K)
                  N=0
                  CALL OPN(N,OP,OPTN)
                  OP1=OBACK(K1)
                  N1=0
                  CALL OPN(N1,OP,OPTN)
                  IF((N .EQ. N1) .OR. (N .EQ. 1)) GOTO 2027
                  GOTO 2025
2027              IF (N .EQ. N1) THEN
                    CLRUDM(C1,L1)=CLRUDM(C1,L1)+TOTCDM(J)
                    DO 427 K4=1,NUMLOC
                       CLDMLC(C1,L1,K4) = CLRUDM(C1,L1)*PCTOP(K4)
427                 CONTINUE
                    IFLG(I)=1
                  ENDIF
                  LFLG(J)=1
                ENDIF
422           CONTINUE
            ENDIF
420       CONTINUE
          IF(LFLG(NMCMOP) .EQ. 0) THEN
            CM=LCOMP(I)
            C1=0
            CALL CMPRTN(C1,CM,COMP)
            LR=LRU(I)
            L1=0
            CALL LRURTN(L1,LR,NLRU)
            CLRUDM(C1,L1)=TOTCDM(NMCMOP)
```

```
         DO 428 K5=1,NUMLOC
            CLDMLC(C1,L1,K5) = CLRUDM(C1,L1)*PCTOP(K5)
428      CONTINUE
       ENDIF
       END
c ****
c * This routine finds an index value for each LRU
c ****

       SUBROUTINE LRURTN(L1,LR,NLRU)
       CHARACTER*10 LR,NLRU(50)
       INTEGER L1
2029   L1=L1+1
       IF(LR .NE. NLRU(L1)) GOTO 2029
       END
c ****
c * This routine takes the repair list by LRU and builds a
c * complete enumeration list for all sets of choices
c ****
       SUBROUTINE LSTREP(NUMLRU,NUMREP,RCNT)

       INTEGER RCNT,NUMLRU,RPL(20),RFLG,NUMREP(20)
       CHARACTER*4 LRUREP(5000,10),REPLST(20,20)
       COMMON /RPBLK/ LRUREP,REPLST
c ****
c * RCNT represents the total number of enumerated choices
c ****
       RCNT=1
       DO 440 I=1,NUMLRU
         RPL(I)=NUMREP(I)
         RCNT=RCNT*NUMREP(I)
440    CONTINUE

       I=1
       J=NUMLRU
       K=NUMLRU-1
       RFLG=0
c ****
c * This routine is similar to the ENUM routine
c ****
2031   IF(RFLG .EQ. 0) THEN
         DO 450 J5=1,NUMREP(J)
2033       LRUREP(I,J)=REPLST(J,RPL(J))
           J=J-1
           IF(J .NE. 0) GOTO 2033
           I=I+1
           J=NUMLRU
           RPL(J)=RPL(J) - 1
450      CONTINUE
       ENDIF
```

```
      RPL(J)=NUMREP(J)
      RPL(K)=RPL(K)-1
      IF((RFLG .EQ. 1) .AND. (RPL(K) .NE. 0)) THEN
        IF(K .EQ. 1) THEN
          K=NUMLRU-1
        ELSE
          K=K+1
        ENDIF
      ENDIF
      RFLG=0
      IF(RPL(K) .EQ. 0) THEN
        RPL(K)=NUMREP(K)
        RFLG=1
        K=K-1
      ENDIF
      IF(K .NE. 0) GOTO 2031
      END
c ****
c * This routine performs the same operations as the above
c * routine for the production locations
c ****
      SUBROUTINE LSTPRD(NUMLRU,LPROD,PCNT)

      INTEGER PCNT,NUMLRU,PRD(20),PFLG,LPROD(25)
      INTEGER LPRDCL(10,10),LRUPRD(5000,10)
      COMMON /PRDBLK/ LRUPRD,LPRDCL
c ****
c * PCNT is the number of production choices
c ****
      PCNT=1
      DO 460 I=1,NUMLRU
        PRD(I)=LPROD(I)
        PCNT=PCNT*LPROD(I)
460   CONTINUE

      I=1
      J=NUMLRU
      K=NUMLRU-1
      PFLG=0

2035  IF(PFLG .EQ. 0) THEN
        DO 470 J5=1,LPROD(J)
2037      LRUPRD(I,J)=LPRDCL(J,PRD(J))
          J=J-1
          IF(J .NE. 0) GOTO 2037
          I=I+1
          J=NUMLRU
          PRD(J)=PRD(J) - 1
470     CONTINUE
      ENDIF
```

```
      PRD(J)=LPROD(J)
      PRD(K)=PRD(K)-1
      IF((PFLG .EQ. 1) .AND. (PRD(K) .NE. 0)) THEN
        IF(K .EQ. 1) THEN
          K=NUMLRU-1
        ELSE
          K=K+1
        ENDIF
      ENDIF
      PFLG=0
      IF(PRD(K) .EQ. 0) THEN
        PRD(K)=LPROD(K)
        PFLG=1
        K=K-1
      ENDIF
      IF(K .NE. 0) GOTO 2035
      END
c ****
c * This routine computes the production batch size and
c * reorder point for each component at the manufacturer's
c ****
      SUBROUTINE LEADTM(NMCOMP,NUMLRU,CLRUDM,CSTORD,HLDCST,
     +   CSTPUR,PDCM,SETPTM,CYCLTM,CELOAD,LT,PRDCEL,TMUNIT,
     +   CELCAP,AVCWT,EOQ,ROP,CSTSET,CSTLAB,PC,MIC)
      INTEGER NUMLRU,PDCM(50),CELL,PRDCEL(50,10),TMUNIT
      REAL CLRUDM(50,25),CSTORD(25),HLDCST(10),CSTPUR(50)
      REAL SETPTM(50,7),CYCLTM(50,7),LT(50),WAIT,EQ(50)
      REAL CELOAD(25),TCD(50),CELCAP(25),AVCWT(50),EOQ(50)
      REAL ROP(50),CP,CH,DRATE,XMU,SIGMA,ACWT,MIC(50)
      REAL CSTSET(50,7),CSTLAB(25),PC(50)
      DATA (TCD(I), I=1,50)/50*0.0/
      DATA (EQ(I), I=1,50)/50*0.0/

      DO 475 I=1,NMCOMP
        DO 480 J=1,NUMLRU
          TCD(I)=TCD(I)+CLRUDM(I,J)
480     CONTINUE
          DRATE = TCD(I)/TMUNIT
          ACWT= AVCWT(I)
          CELL=PDCM(I)
          CP = CSTORD(CELL)
          CH = HLDCST(CELL)*CSTPUR(I)
          L10=0
          CALL CELRTN(I,L10,CELL,PRDCEL)
          EQ(I)=SQRT((DRATE*2.0*CP)/CH)
          WAIT=LOG(CELOAD(L10))
          LT(I)=((SETPTM(I,L10)+CYCLTM(I,L10)*EQ(I)+WAIT)/
     +              CELCAP(L10))
          XMU=LT(I)*DRATE
          SIGMA= SQRT(LT(I))*DRATE
```

```
          IF(SIGMA .LE. 1.0) THEN
            SIGMA=1.0
          ENDIF
c ****
c * Passes; the wait time, demand rate during lead time,
c * standard deviation, order cost and holding cost
c ****
          CALL HW(Q,R,ACWT,DRATE,XMU,SIGMA,CP,CH)
          EOQ(I)=Q
          ROP(I)=R
c ****
c * Receives the batch size and order point, then computes
c * the production cost and inventory cost at manufacturer
c ****
          PC(I)=(DRATE/EOQ(I)*CSTSET(I,L10)+
     +              DRATE*CSTLAB(CELL))*TMUNIT
          SS=R-XMU
          IF(SS .LT. 0) THEN
            MIC(I)=0
          ELSE
            MIC(I)=SS*CH*TMUNIT
          ENDIF
          TCD(I)=0.0
475   CONTINUE
      END
c ****
c * Finds an index value for the production cells
c ****
      SUBROUTINE CELRTN(I,L,CELL,PRDCEL)
      INTEGER CELL,PRDCEL(50,10)
2039  L=L+1
      IF(PRDCEL(I,L) .NE. CELL) GOTO 2039
      END

c ****
c * Performs the same set of operations as the above routine
c * for each standby option
c ****
      SUBROUTINE SLDTM(CNT,TOTSDM,PDCM,CSTORD,HLDCST,CSTPUR,
     +    SETPTM,CYCLTM,CELOAD,SLT,PRDCEL,TMUNIT,CELCAP,
     +    NMOPTN,OPTIND,COMP,COMPNM,OPRQMT,ENM,SEOQ,SROP,
     +    AVCWT,SPC,CSTSET,CSTLAB,SMIC)

      INTEGER CNT(25),PDCM(50),OPTIND(50),CELL,PRDCEL(50,10)
      INTEGER OPRQMT(50),C1,OP,ENM(5,10,25),TMUNIT
      REAL TOTSDM(25,50),CSTORD(25),HLDCST(10),CSTPUR(50)
      REALSETPTM(50,7),CYCLTM(50,7),SLT(10,50),SWAIT,SEQ(50)
     REAL CELOAD(25),CELCAP(25),SEOQ(10,25,50),SROP(10,25,50)
      REAL AVCWT(50),SPC(10,50),CSTSET(50,7),CSTLAB(25)
      REAL SMIC(10,50),LT
```

```
      CHARACTER*15 COMP(50),COMPNM(50),CM

      DO 490 I=1,NMOPTN
        IF (OPTIND(I) .EQ. 3) THEN
          DO 495 J=1,CNT(I)
            LT=0
            DO 497 K=1,OPRQMT(I)
              OP=ENM(I,J,K)
              C1=0
              CM=COMPNM(OP-NMOPTN)
              CALL CMPRTN(C1,CM,COMP)
              CELL=PDCM(C1)
              ACWT=AVCWT(C1)
              DRATE=TOTSDM(I,J)/TMUNIT
              CP=CSTORD(CELL)
              CH=HLDCST(CELL)*CSTPUR(C1)
              L=0
              CALL CELRTN(C1,L,CELL,PRDCEL)
              SEQ(K)=SQRT(DRATE*2.0*CP/CH)
              SWAIT=LOG(CELOAD(L))
              LT = LT+((SETPTM(C1,L)+CYCLTM(C1,L)*SEQ(K)
     +                  +SWAIT)/CELCAP(L))
              XMU=DRATE*LT
              SIGMA=DRATE*SQRT(LT)
              IF(SIGMA .LE. 1.0) THEN
                SIGMA=1.0
              ENDIF
              CALL HW(Q,R,ACWT,DRATE,XMU,SIGMA,CP,CH)
              SEOQ(I,J,C1)=Q
              SROP(I,J,C1)=R
              SPC(I,J)=SPC(I,J)+(DRATE/SEOQ(I,J,C1)*
     +              CSTSET(C1,L)+DRATE*CSTLAB(CELL))*TMUNIT
              SS=R-XMU
              IF(SS .LT. 0) THEN
                SMIC(I,J)=0
              ELSE
                SMIC(I,J)=SS*CH*TMUNIT
              ENDIF
497         CONTINUE
            SLT(I,J)=LT
            SPC(I,J)=0
495       CONTINUE
        ENDIF
490     CONTINUE
        END


    SUBROUTINE HW(QSTAR,RSTAR,ACWT,DRATE,XMU,SIGMA,CP,CH)
      COMMON/PROB/X,RTAIL,ALPHA,BETA
```

```
c ****
c * FIRST WE TRANSLATE THE CONSTRAINT ON ACWT INTO A
c * CONSTRAINT ON B
c ****
      BLIM=ACWT*DRATE
      EPS=1.
c ****
C * NOW WE COMPUTE THE EOQ AND ROP USING SIMPLE HEURISTIC
c ****
      Q=SQRT(2.*DRATE*CP/CH)
      Q=FLOAT(INT(Q+.5))
      CALL ROP(R,BLIM,XMU,Q,SIGMA)
      RSTAR=R
      QSTAR=Q
      ICOUNT=1
c ****
c *  NOW WE DO HADLEY-WHITIN APPROACH
c ****
  525 X=(R+Q-XMU)/SIGMA
      CALL NORMAL
      HOLD=SIGMA*ALPHA
      X=(R-XMU)/SIGMA
      CALL NORMAL
      IF(ALPHA.GT.0001)THEN
      HOLD=2.*(BLIM-HOLD)/(SIGMA*ALPHA-HOLD)
      ELSE
      HOLD=0.
      ENDIF
      HOLD=1.-HOLD
      QSTAR=SQRT((2.*DRATE*CP)/(CH*HOLD))
      QSTAR=FLOAT(INT(QSTAR+.5))
      CALL ROP(RSTAR,BLIM,XMU,QSTAR,SIGMA)
c ****
c * NOW WE ITERATE TO GET SOLUTION
c ****
      IF((ABS(Q-QSTAR).LT.EPS).AND.
     +       (ABS(R-RSTAR).LT.EPS))RETURN
      IF(ICOUNT.GT.20)RETURN
      Q=QSTAR
      R=RSTAR
      ICOUNT=ICOUNT+1
      GO TO 525
      STOP
      END


      SUBROUTINE ROP(Y,CUT,XMU,Q,SIGMA)
      COMMON/PROB/X,RTAIL,ALPHA,BETA
```

```
C ****
C * THIS SUBROUTINE TAKES A CUTOFF VALUE AND FINDS THE REORDER
C *   POINT THAT SATIFIES THE CUTOFF VIA THE NEWSBOY MODEL
C ****
      Y=FLOAT(INT(XMU))
      HOLD=Y
      EPS=1.
C ****
C *  FIRST WE DO A BOUNDING SEARCH
C ****
      ICOUNT=0
  510 X=(Y-XMU)/SIGMA
      CALL NORMAL
      B=BETA
      X=(Y+Q-XMU)/SIGMA
      CALL NORMAL
      B=B-BETA
      B=SIGMA**2*B/Q
      DIFF=CUT-B
      IF(DIFF.GT.0.)THEN
      SIGN=1.
      ELSE
      SIGN=-1.
      ENDIF
      IF(ICOUNT.GT.0)THEN
      IF(DRCTN.NE.SIGN)GO TO 520
C     IF((X.GT.3.).OR.(X.LT.-3.))GO TO 520
      IF(ICOUNT.GT.3)GO TO 520
      IF(ABS(DIFF).LE.(.01*CUT))RETURN
      ENDIF
      DRCTN=SIGN
      Y=Y-DRCTN*SIGMA
      Y=FLOAT(INT(Y+.5))
      ICOUNT=ICOUNT+1
      GO TO 510
C ****
C *  NOW WE REFINE ESTIMATE
C ****
  520 Y1=AMIN1(Y,HOLD)
      Y2=AMAX1(Y,HOLD)
      ICOUNT=1
  540 Y=(Y1+Y2)*.5
      Y=FLOAT(INT(Y+.5))
      X=(Y-XMU)/SIGMA
      CALL NORMAL
      B=BETA
      X=(Y+Q-XMU)/SIGMA
      CALL NORMAL
      B=B-BETA
      B=SIGMA**2*B/Q
```

```
      DIFF=CUT-B
      IF(ABS(DIFF).LT.(.01*CUT))RETURN
      IF((Y2-Y1).LT.EPS)RETURN
      IF(DIFF.GT.0.)THEN
      Y2=Y
      ELSE
      Y1=Y
      ENDIF
      IF(ICOUNT.GT.20.)RETURN
      ICOUNT=ICOUNT+1
      GO TO 540
      END

      SUBROUTINE NORMAL
      COMMON/PROB/X,RTAIL,ALPHA,BETA
      FALPHA(X)=EXP(-.5*X*X)/2.507-X*FRTAIL(X)
      FBETA(X)=(.5+.5*X*X)*FRTAIL(X)-X*EXP(-.5*X*X)/5.013
C ****
C * THIS SUBROUTINE COMPUTES NORMAL PROBABILITIES, RIGHT-HAND
C * TAILS, AND ALPHAS
C
      ZSIG=3.
      IF(X.LT.-ZSIG)GO TO 540
      IF(X.GT.ZSIG)GO TO 550
      RTAIL=FRTAIL(X)
      ALPHA=FALPHA(X)
      BETA=FBETA(X)
      RETURN
  540 DENS=0.
      RTAIL=1.
      ALPHA=-X
      BETA=.5+.5*X*X
      RETURN
  550 DENS=0.
      RTAIL=0.
      ALPHA=0.
      BETA=0.
      RETURN
      END
      FUNCTION FRTAIL(Z)
C ****
C *    THIS SUBROUTINE COMPUTES NORMAL RIGHT-HAND TAILS
C ****
      IFLAG=0
      IF(Z.LE.0)THEN
      IFLAG=1
      Z=-Z
      ENDIF
      T=1./(1.+.33267*Z)
      P2=1./SQRT(6.28318)*EXP(-Z**2/2)
```

```
      P2=P2*(.4361836*T-.1201676*T**2+.9372980*T**3)
      IF(IFLAG.NE.0)THEN
      FRTAIL=1.-P2
      Z=-Z
      RETURN
      ENDIF
      FRTAIL=P2
      RETURN
      END

c ****
c * This routine is similar to the LEADTM routine, it computes
c * the inventory cost by location and calls HW to compute
c * the order size.  It also computes transportation cost
c ****
      SUBROUTINE FLDINV(NUMLOC,CLDMLC,NMCOMP,NUMLRU,RPCM,
     +     DEPOT,TRANTM,AVCWT,CSTCHD,CSTPUR,RPORD,TOTFIC,LT,
     +     TMUNIT,TRNCST,TOTXC,DMNDRT,COMPNM,COMP,AVAIL,
     +     NMCMOP,ORD,LOP)

      INTEGER NUMLOC,NUMLRU,NMCOMP,D1,TMUNIT,C1
      REALCLDMLC(50,25,20),TRANTM(15,5),AVCWT(50),CSTCHD(60)
      REAL CSTPUR(50),RPORD(20),TOTFIC(50),SS(10,50),LT(50)
      REAL TCD(50),FIC(10,50),LDTM,ORD(10,50),LOP(10,50)
      REAL TRNCST(15,5),XC(10,50),TOTXC(50),DMNDRT(50)
      REAL AVAIL(50,10),MTBF,DWNTM,COMPRT(50)
      CHARACTER*4 DEPOT(15,5),RPCM(50),DP
      CHARACTER*15 COMPNM(50),COMP(50),CM
      DATA (TCD(I),I=1,50)/50*0.0/
      DATA ((FIC(I,J),I=1,10),J=1,50)/500*0.0/
      DATA ((SS(I,J),I=1,10),J=1,50)/500*0.0/

      DO 557 I=1,50
        TOTFIC(I)=0
        TOTXC(I)=0
557   CONTINUE

c ****
c * Gets the component demand rate
c ****

      DO 780 I=1,NMCMOP
        C1=0
        CM=COMPNM(I)
        CALL CMPRTN(C1,CM,COMP)
        COMPRT(C1)=DMNDRT(I)
780   CONTINUE
```

```
c ****
c * Computes total component demand per LRU at each location
c ****

      DO 560 I=1,NUMLOC
        DO 565 J=1,NMCOMP
          DO 568 K=1,NUMLRU
            TCD(J)=TCD(J)+CLDMLC(J,K,I)
568       CONTINUE
c ****
c * Checks where repair is taking place, 0 is the manufacturer
c * location.  If repair is at the operating location, then
c * transportation time includes going to the depot also.
c ****
        D1=0
        DP='   0'
        CALL DEPRTN(I,D1,DP,DEPOT)
        LDTM=TRANTM(I,D1)/250
        COST=TRNCST(I,D1)
        IF((RPCM(J) .EQ. 'OPLC') .OR.
     +      (RPCM(J) .EQ. '   0'))THEN
          LDTM=TRANTM(I,D1)/250
          COST=TRNCST(I,D1)
        ELSE
          DP=RPCM(J)
          D1=0
          CALL DEPRTN(I,D1,DP,DEPOT)
          LDTM=(TRANTM(I,D1)+LDTM)/250
          COST=COST+TRNCST(I,D1)
        ENDIF
        LDTM=LDTM+LT(J)
        DRATE=TCD(J)/TMUNIT
        ACWT=AVCWT(J)
        CH=CSTCHD(I)*CSTPUR(J)
        CP=RPORD(I)
        XMU=DRATE*LDTM
        SIGMA=DRATE*SQRT(LDTM)
        IF(SIGMA .LE. 1.0) THEN
          SIGMA=1.0
        ENDIF
        CALL HW(Q,R,ACWT,DRATE,XMU,SIGMA,CP,CH)
        ORD(I,J)=Q
        LOP(I,J)=R
        SS(I,J)=R-XMU
        IF(SS(I,J) .LT. 0) THEN
          FIC(I,J)=0.0
        ELSE
          FIC(I,J)=SS(I,J)*CSTPUR(J)*CSTCHD(I)
        ENDIF
        TRIPS=DRATE/Q
```

```
          XC(I,J)=TRIPS*COST
          DWNTM=LDTM*250.0*8.0
          MTBF=1/COMPRT(J)
          AVAIL(J,I)=MTBF/(MTBF+DWNTM)
          COST=0.0
          TCD(J)=0.0
          LDTM=0.0
565       CONTINUE
560     CONTINUE
        DO 570 L=1,NMCOMP
          DO 575 L1=1,NUMLOC
            TOTFIC(L)=FIC(L1,L)*TMUNIT+TOTFIC(L)
            TOTXC(L)=XC(L1,L)*TMUNIT+TOTXC(L)
575       CONTINUE
570     CONTINUE
        END
c ****
c * This routine performs the same operations as the above
c * routine for standby assemblies
c ****
        SUBROUTINE STNINV(NUMLOC,SDMDLC,NMOPTN,RPLC,OPTIND,
       +        DEPOT,TRANTM,AVCWT,CSTCHD,CSTPUR,RPORD,TOTSIC,
       +        SLT,TMUNIT,TRNCST,TOTSXC,COMP,LRU,COMPNM,NLRU,
       +        CNT,OPRQMT,ENM,RT,SAVAIL,SORDLC,SROPLC)

        INTEGER NUMLOC,D1,TMUNIT,C1,OP,OPTIND(50),NMOPTN
        INTEGER CNT(25),OPRQMT(50),ENM(5,10,25)
        REALSDMDLC(20,20,50),TRANTM(15,5),AVCWT(50),CSTCHD(60)
        REAL CSTPUR(50),RPORD(20),TOTSIC(10,50),SS(10,10,50)
        REAL FIC(10,10,50),SORDLC(10,10,50),SROPLC(10,10,50)
        REALTRNCST(15,5),XC(10,10,50),TOTSXC(10,50),SLT(10,50)
        REAL RT(10,50),SAVAIL(10,10,50),MTBF,DWNTM,,LDTM
        CHARACTER*4 DEPOT(15,5),RPLC(50),DP
        CHARACTER*15 CM,COMP(50),COMPNM(50)
        CHARACTER*10 LR,LRU(50),NLRU(50)
        DATA (((FIC(I,J,K),I=1,10),J=1,10),K=1,50)/5000*0.0/
        DATA (((SS(I,J,K),I=1,10),J=1,10),K=1,50)/5000*0.0/

c ****
c * Initializes the values
c ****

        DO 625 I=1,10
          DO 630 J=1,50
            TOTSIC(I,J)=0.0
            TOTSXC(I,J)=0.0
630       CONTINUE
625     CONTINUE
```

```
      DO 670 I=1,10
        DO 672 J=1,10
          DO 675 K=1,50
            FIC(I,J,K)=0.0
            XC(I,J,K)=0.0
            SS(I,J,K)=0.0
675       CONTINUE
672     CONTINUE
670   CONTINUE


      DO 635 I=1,NMOPTN
       IF (OPTIND(I) .EQ. 3) THEN
        DO 640 J=1,NUMLOC
         DO 645 K=1,CNT(I)
          DO 647 L=1,OPRQMT(I)
          OP=ENM(I,K,L)
          L1=0
          LR=LRU(OP-NMOPTN)
          CALL LRURTN(L1,LR,NLRU)
          D1=0
          DP='   0'
          CALL DEPRTN(L1,D1,DP,DEPOT)
          LDTM=TRANTM(J,D1)/250
          COST=TRNCST(J,D1)
          IF((RPLC(L1).EQ.'OPLC') .OR.
     +       (RPLC(L1).EQ.'   0'))THEN
            LDTM=TRANTM(J,D1)/250
            COST=TRNCST(J,D1)
          ELSE
            DP=RPLC(L1)
            D1=0
            CALL DEPRTN(L1,D1,DP,DEPOT)
            LDTM=TRANTM(J,D1)/250+LDTM
            COST=COST+TRNCST(J,D1)
          ENDIF
          LDTM=LDTM+SLT(I,K)
          C1=0
          CM=COMPNM(OP-NMOPTN)
          CALL CMPRTN(C1,CM,COMP)
          DRATE=SDMDLC(I,K,J)/TMUNIT
          ACWT=AVCWT(C1)
          CH=CSTCHD(J)*CSTPUR(C1)
          CP=RPORD(J)
          XMU=DRATE*LDTM
          SIGMA=DRATE*SQRT(LDTM)
          IF(SIGMA .LE. 1.0) THEN
           SIGMA=1.0
          ENDIF
          CALL HW(Q,R,ACWT,DRATE,XMU,SIGMA,CP,CH)
```

```
            SORDLC(I,J,K)=Q
            SROPLC(I,J,K)=R
            SS(I,J,K)=R-XMU
            IF(SS(I,J,K) .LT. 0) THEN
              FIC(I,J,K)=FIC(I,J,K)
            ELSE
              FIC(I,J,K)=SS(I,J,K)*CSTPUR(C1)*CSTCHD(J)
     +                  +FIC(I,J,K)
            ENDIF
            TRIPS=DRATE/Q
            XC(I,J,K)=TRIPS*COST+XC(I,J,K)
            DWNTM=LDTM*250.0*8.0
            MTBF=1/RT(I,K)
            SAVAIL(I,J,K)=MTBF/(MTBF+DWNTM)
            COST=0.0
            LDTM=0.0
647         CONTINUE
645       CONTINUE
640      CONTINUE
         ENDIF
635     CONTINUE
        DO 650 I=1,NMOPTN
         IF(OPTIND(I) .EQ. 3) THEN
          DO 652 J=1,CNT(I)
           DO 655 L1=1,NUMLOC
             TOTSIC(I,J)=FIC(I,L1,J)*TMUNIT+TOTSIC(I,J)
             TOTSXC(I,J)=XC(I,L1,J)*TMUNIT+TOTSXC(I,J)
655        CONTINUE
652       CONTINUE
         ENDIF
650     CONTINUE
        END

c ****
c * Finds an index value for each depot location
c ****
        SUBROUTINE DEPRTN(I,D1,DP,DEPOT)
        CHARACTER*4 DEPOT(15,5),DP
        INTEGER D1
2041    D1=D1+1
        IF(DEPOT(I,D1) .NE. DP) GOTO 2041
        END

c ****
c * Computes the repair cost
c ****
       SUBROUTINE REPCST(CSTRPL,COMTST,LRUTST,LRURPL,CMPRMR,
      +           RPLC,LRU,NLRU,REPLOC,COMP,NUMLOC,NMCMOP,
      +           NMCOMP,TOTRPC,REPLD,TMUNIT,LCOMP,
      +           NUMLRU,CDMDLC)
```

```
      INTEGER NMCMOP,NMCOMP,R1,C1,TMUNIT
      REAL CSTRPL(60),COMTST(50),LRUTST(10,15),LRURPL(50,25)
      REAL CMPRMR(50),TOTRPC(50),REPLD(50)
      REALRPRTM(10,50),RPRCST(10,50),CDMDLC(20,50),NMREP(50)
      CHARACTER*4 RPLC(50),RP,REPLOC(20,50)
      CHARACTER*10 LRU(50),NLRU(50),LR
      CHARACTER*15 COMP(50),CM,LCOMP(50)

      DO 583 I=1,NUMLOC
        DO 584 J=1,NMCOMP
          RPRTM(I,J)=0.0
          RPRCST(I,J)=0.0
584     CONTINUE
583   CONTINUE

      DO 588 I=1,NMCMOP
        TOTRPC(I)=0.0
        NMREP(I)=0.0
588   CONTINUE

      DO 580 I=1,NUMLOC
        DO 582 J=1,NMCMOP
          LR=LRU(J)
          L1=0
          CALL LRURTN(L1,LR,NLRU)
          R1=0
c ****
c * Gets the repair location from the middle problem
c ***
          RP=RPLC(L1)
          IF(RP .EQ. 'OPLC') THEN
            R1=I
          ELSE
            CALL REPRTN(L1,R1,RP,REPLOC)
          ENDIF
          CM=LCOMP(J)
          C1=0
          CALL CMPRTN(C1,CM,COMP)
          WAIT=LOG(REPLD(I))
c ***
c * Repair time is the test time and remove and replace time
c ****
          RPRTM(I,C1)=LRUTST(L1,R1)+LRURPL(L1,R1)+COMTST(J)+
     +                CMPRMR(J)+RPRTM(I,C1)+WAIT
c ****
c * Computes Repair Cost
          RPRCST(I,C1)=RPRTM(I,C1)*CSTRPL(I)*(CDMDLC(I,J)/
     +                 TMUNIT)
582       CONTINUE
580   CONTINUE
```

```
      DO 585 I=1,NMCOMP
        DO 587 J=1,NUMLOC
          TOTRPC(I)=RPRCST(J,I)*TMUNIT+TOTRPC(I)
587     CONTINUE
585   CONTINUE
      END

c ****
c * Performs the same operations for the standby assemblies
c ****

      SUBROUTINE STNREP(CSTRPL,COMTST,LRUTST,LRURPL,CMPRMR,
     +           RPLC,LRU,NLRU,REPLOC,NUMLOC,OPTIND,ENM,
     +           NMOPTN,TOTSRP,REPLD,TMUNIT,CNT,
     +           OPRQMT,SDMDLC)
      INTEGER OPRQMT(50),R1,OP,TMUNIT,CNT(25),ENM(5,10,25)
      INTEGER OPTIND(50)
      REAL CSTRPL(60),COMTST(50),LRUTST(10,15),LRURPL(50,25)
      REAL CMPRMR(50),TOTSRP(10,50),REPLD(50)
      REAL RPTM,RPCST(10,20,50),SDMDLC(20,20,50)
      CHARACTER*4 RPLC(50),RP,REPLOC(20,50)
      CHARACTER*10 LRU(50),NLRU(50),LR
c ****
c * Initializes values
c ****

      DO 720 I=1,10
        DO 722 J=1,50
          TOTSRP(I,J)=0.0
722     CONTINUE
720   CONTINUE

      DO 724 I=1,10
       DO 726 J=1,20
        DO 728 K=1,50
         RPCST(I,J,K)=0.0
728     CONTINUE
726    CONTINUE
724   CONTINUE

      RPTM=0.0
      DO 700 I=1,NMOPTN
       IF(OPTIND(I) .EQ. 3) THEN
        DO 702 J=1,NUMLOC
         DO 704 K=1,CNT(I)
          DO 706 L=1,OPRQMT(I)
            OP=ENM(I,K,L)
            LR=LRU(OP-NMOPTN)
            L1=0
            CALL LRURTN(L1,LR,NLRU)
```

```
            R1=0
c ****
c * Gets repair location from middle problem
c ****
            RP=RPLC(L1)
            IF(RP .EQ. 'OPLC') THEN
              R1=I
            ELSE
              CALL REPRTN(L1,R1,RP,REPLOC)
            ENDIF
            WAIT=LOG(REPLD(J))
c ****
c * Computes repair time
c ****
            RPTM=LRUTST(L1,R1)+LRURPL(L1,R1)+
     +       COMTST(OP-NMOPTN)+CMPRMR(OP-NMOPTN)+RPTM+WAIT
706         CONTINUE
          RPCST(I,J,K)=(RPTM*CSTRPL(J))*
     +                  (SDMDLC(I,K,J)/TMUNIT)
          RPTM=0.0
704       CONTINUE
702      CONTINUE
        ENDIF
700   CONTINUE

      DO 708 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 3) THEN
          DO 710 J=1,CNT(I)
            DO 712 K=1,NUMLOC
             TOTSRP(I,J)=RPCST(I,K,J)*TMUNIT+TOTSRP(I,J)
712         CONTINUE
710       CONTINUE
        ENDIF
708   CONTINUE
      END
c ****
c * Finds an index value for the repair locations
c ****
      SUBROUTINE REPRTN(L1,R1,RP,REPLOC)
      CHARACTER*4 RP,REPLOC(20,50)
      INTEGER L1,R1
2043  R1=R1+1
      IF(REPLOC(L1,R1) .NE. RP) GOTO 2043
      END
c ****
c * Computes the disposal cost for the LRU
c ****
      SUBROUTINE DISPOS(NMCOMP,NUMLRU,TMUNIT,CNDMRT,CSTDIS,
     +           NMOPTN,OPTIND,CNT,DISCST,ENM,LRU,NLRU,
     +           SDISCS,CLRUDM,TOTSDM)
```

```
      INTEGER NMCOMP,NUMLRU,OPTIND(50),CNT(25),ENM(5,10,25)
      INTEGER TMUNIT,NMOPTN,OP
      REAL CNDMRT(20),CSTDIS(20),DISCST(50,10),SDISCS(10,50)
      REAL CLRUDM(50,25),TOTSDM(25,50),NMDIS,SDIS
      CHARACTER*10 LRU(50),NLRU(50),LR
c ****
c * Computes expected number that will be condemned based on
c * the demand rate and multiplies by cost
c ****
      DO 740 I=1,NMCOMP
        DO 742 J=1,NUMLRU
          NMDIS=(CLRUDM(I,J)/TMUNIT)*CNDMRT(J)
          DISCST(I,J)=NMDIS*CSTDIS(J)*TMUNIT
742     CONTINUE
740   CONTINUE
c ****
c * For the standby assemblies
c ****
      DO 744 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 3) THEN
          DO 746 J=1,CNT(I)
            OP=ENM(I,J,1)
            L1=0
            LR=LRU(OP-NMOPTN)
            CALL LRURTN(L1,LR,NLRU)
            SDIS=(TOTSDM(I,J)/TMUNIT)*CNDMRT(L1)
            SDISCS(I,J)=SDIS*CSTDIS(L1)*TMUNIT
746       CONTINUE
        ENDIF
744   CONTINUE
      END

c ****
c * This routine compares the configuration options for
c * each component requirement to determine if any are
c * dominated based on LCC and availability
c ****

      SUBROUTINE DMNATE(NUMLOC,NMOPTN,COMIND,OPTIND,CNT,
     +     OPRQLS,OPRQMT,COMPNM,COMP,NUMOP,RQMT,DSFLG,DFLG,
     +     AVAIL,SAVAIL,ACQCST,TOTXC,TOTFIC,MIC,PC,TOTSXC,
     +     TOTSIC,SMIC,SPC,TOTRPC,TOTSRP,DISCST,SDISCS,LCC,
     +   SLCC,RQMTOP,NMCOMP,NUMLRU,ENM,NMRQMT,SAVGAV,AVGAVL)

      INTEGER NUMLOC,NMOPTN,COMIND(25),OPTIND(50),OPRQMT(50)
      INTEGER NUMOP(50),DSFLG(5,25),DFLG(50),R1,C1,OP,
      INTEGER CNT(25),ENM(5,10,25),OP2,C2,RQMTOP(30,5)
      REAL AVAIL(50,10),SAVAIL(10,10,50),ACQCST(50),TOTRPC(50)
      REAL TOTSRP(10,50),DISCST(50,10),SDISCS(10,50),TOTXC(50)
      REAL TOTFIC(50),MIC(50),PC(50),TOTSXC(10,50)
```

```
      REAL TOTSIC(10,50),SACQCS(10,50),AVGAVL(50)
      REAL SMIC(10,50),SPC(10,50),LCC(50),SLCC(10,50)
      REAL DIS(50),TMPAVL(50),TMP(10,50),SAVGAV(10,50)
      CHARACTER*3 OPRQLS(50,25),RQMT(25),RQ
      CHARACTER*15 COMPNM(50),COMP(50),CM,CM2

      DO 800 I=1,10
        DO 802 J=1,50
          SACQCS(I,J)=0.0
          TMP(I,J)=0.0
802     CONTINUE
800   CONTINUE

      DO 1300 I=1,5
        DO 1302 J=1,25
          DSFLG(I,J)=0
1302    CONTINUE
1300  CONTINUE

      DO 804 I=1,50
        DIS(I)=0.0
        TMPAVL(I)=0.0
        DFLG(I)=0
804   CONTINUE

      DO 806 I=1,NMCOMP
        DO 808 J=1,NUMLRU
          DIS(I)=DISCST(I,J)+DIS(I)
808     CONTINUE
806   CONTINUE
c ****
c * Computes avaverage availability and LCC for the standby
c * assemblies
c ****
      DO 810 I=1,NMOPTN
        IF(OPTIND(I) .EQ. 3) THEN
          DO 812 J=1,CNT(I)
            DO 814 K=1,OPRQMT(I)
              OP=ENM(I,J,K)
              SACQCS(I,J)=SACQCS(I,J)+ACQCST(OP-NMOPTN)
814         CONTINUE
          SLCC(I,J)=TOTSXC(I,J)+TOTSIC(I,J)+SMIC(I,J)+
     +          TOTSRP(I,J)+SACQCS(I,J)+SDISCS(I,J)+SPC(I,J)
          DO 822 L=1,NUMLOC
          TMP(I,J)=SAVAIL(I,L,J)+TMP(I,J)
822       CONTINUE
          SAVGAV(I,J)=TMP(I,J)/NUMLOC
812       CONTINUE
        ENDIF
810   CONTINUE
```

```
c ****
c * Computes LCC and average availability for all other
c * component options
c ****
      DO 816 I=1,NMCOMP
        LCC(I)=TOTXC(I)+TOTFIC(I)+MIC(I)+TOTRPC(I)+
     +         DIS(I)+ACQCST(I)+PC(I)
        DO 818 J=1,NUMLOC
          TMPAVL(I)=TMPAVL(I)+AVAIL(I,J)
818     CONTINUE
      AVGAVL(I)=TMPAVL(I)/NUMLOC
      WRITE(20,*) I,LCC(I),AVGAVL(I)
816   CONTINUE
c ****
c * If an option is dominated, this routine sets a flag.
c * This is for the standby assemblies
c ****
      DO 820 I=1,NMOPTN
        IF (OPTIND(I) .EQ. 3) THEN
          DO 824 J=1,CNT(I)-1
            DO 826 K=J+1,CNT(I)
              IF ((SLCC(I,J) .GT. SLCC(I,K)) .AND.
     +                (SAVGAV(I,J) .LT. SAVGAV(I,K))) THEN
                DSFLG(I,J)=1
              ENDIF
              IF ((SLCC(I,K) .GT. SLCC(I,J)) .AND.
     +                 (SAVGAV(I,K) .LT. SAVGAV(I,J))) THEN
                DSFLG(I,K)=1
              ENDIF
826         CONTINUE
824       CONTINUE
        ENDIF
820   CONTINUE
c ****
c * This routine is for all other component options.  It also
c * sets a flag if an option is dominated.
c ****
      DO 828 I=1,NMRQMT
        IF(COMIND(I) .EQ. 1) THEN
            RQ=RQMT(I)
            R1=0
            CALL RQR(R1,RQ,RQMT)
            DO 832 K=1,NUMOP(R1)-1
              OP=RQMTOP(R1,K)
              CM=COMPNM(OP-NMOPTN)
              C1=0
              CALL CMPRTN(C1,CM,COMP)
              DO 834 L=K+1,NUMOP(R1)
                C2=0
                OP2=RQMTOP(R1,L)
```

```
                CM2=COMPNM(OP2-NMOPTN)
                CALL CMPRTN(C2,CM2,COMP)
                IF((LCC(C1) .GT. LCC(C2)) .AND.
     +              (AVGAVL(C1) .LT. AVGAVL(C2))) THEN
                  DFLG(OP)=1
                ENDIF
                IF((LCC(C2) .GT. LCC(C1)) .AND.
     +               (AVGAVL(C2) .LT. AVGAVL(C1))) THEN
                  DFLG(OP2)=1
                ENDIF
834           CONTINUE
832         CONTINUE
          ENDIF
828     CONTINUE
        END

c ****
c * This routine builds the decision rules for the inner
c * problem D.P.
c ****
        SUBROUTINE DECRUL(STAGE,NR,RQMT,COMIND,OPTIND,CNT,OBACK,
     +            DSFLG,DFLG,SLCC,SAVGAV,LCC,AVGAVL,GAMMA,
     +           NMOPTN,NUMOP,RQMTOP,COMPNM,COMP,OPRQLS,LINK,
     +            STGCNT,ENDPT,LSTAR,ASTAR,CHOICE)

        INTEGER NR,COMIND(25),OPTIND(50),CNT(25),OBACK(50),R1
        INTEGER DSFLG(5,25),DFLG(50),OP,SCOMPR(15),CHOICE(35,50)
        INTEGER GAMMA,OP1,OP2,OP3,C1,C2,COMPAR(20),NUMOP(50)
        INTEGER RQMTOP(30,5),STGCNT(25),CHC(35,50),OP5,S1,ST(10)
        INTEGER SCNT,LCNT,LFLG,OFLG(20),OP7,PARE(20),S10,OPT(10)
        INTEGER SCMPR(20),OPT1(10)
        REAL SLCC(10,50),LCC(50),SAVGAV(10,50),AVGAVL(50)
        REAL ENDPT(35,50),LSTAR(35,50),ASTAR(35,50)
        REALEND(35,50),LS(35,50),AS(35,50),ATEMP(20),BTEMP(20)
        REAL LST1,LST2
        CHARACTER*3 STAGE(25),RQMT(25),RQ,OPRQLS(50,25),RQ2
        CHARACTER*3 LINK(35,10),RQ4
        CHARACTER*15 COMPNM(50),COMP(50),CM,CM2
c ****
c * Initializes values
c ****
        DO 875 I=1,20
          OFLG(I)=0
875     CONTINUE

        DO 1875 I=1,25
          STGCNT(I)=0
1875    CONTINUE
        LFLG=0
```

```
      M=0
c ****
c * Travels through the requirements of the product structure
c ****
      DO 840 I=1,NR
        RQ=STAGE(I)
        R1=0
        CALL RQR(R1,RQ,RQMT)
        OP=OBACK(R1)
c ****
c * These are the rules for case I stages described in Ch. 4
c ****
        IF((COMIND(R1) .EQ. 1) .AND. (LFLG .EQ. 0) .AND.
     +     (OFLG(OP) .EQ. 0) .AND. LINK(R1,1) .EQ. '  0') THEN
c ****
c * If the requirement is in series, then sets a flag for the
c * next stage will be a case II
c ****
          IF(OP .EQ. 1) LFLG=1
c ****
c * This routine is for standby assemblies
c ****
          IF(OPTIND(OP) .EQ. 3) THEN
            J2=0
            DO 842 J=1,CNT(OP)
c ****
c * Checks the flag to determine if option was dominated
c ****
              IF(DSFLG(OP,J) .EQ. 0) THEN
                J2=J2+1
                SCOMPR(J2)=J
              ENDIF
842         CONTINUE
c ****
c * Computes only if more than one option to compare
c ****
            IF(J2 .GT. 1) THEN
            DO 844 J3=1,J2-1
              DO 846 K=J3+1,J2
                M=M+1
                T1=(SLCC(OP,SCOMPR(J3))-SLCC(OP,SCOMPR(K)))
     +              /100000
                T2=GAMMA*(SAVGAV(OP,SCOMPR(J3))
     +              -SAVGAV(OP,SCOMPR(K)))
                T3=T1/T2
c ****
c * Endpoint for the decision rule
c ****
```

```
                IF (T3 .GT. 1.0) THEN
                  T3=1.0
                ENDIF
c ****
c * Saves the endpoint, option choice, L-star and A-star
c * values for later comparisons
c ****
                END(I,M)=T3
                CHC(I,M)=SCOMPR(K)
                LS(I,M)=SLCC(OP,SCOMPR(K))/100000
                AS(I,M)=SAVGAV(OP,SCOMPR(K))
                M=M+1
                END(I,M)=T3
                CHC(I,M)=SCOMPR(J3)
                LS(I,M)=SLCC(OP,SCOMPR(J3))/100000
                AS(I,M)=SAVGAV(OP,SCOMPR(J3))
846           CONTINUE
844         CONTINUE
c ****
c * Counts the number of endpoints were computed
c ****
            STGCNT(I)=M
c ****
c * This subroutine sorts the endpoints for comparisons
c * at later stages
c ****
            CALL SRTEND(I,LS,AS,END,CHC,ENDPT,CHOICE,ASTAR,
     +           LSTAR,STGCNT,J2,K20)
            STGCNT(I)=K20
c ****
c * Performs this operation only if there is one option
c * that dominated all of the others
c ****
            ELSE
                ENDPT(I,1)=0.0
                CHOICE(I,1)=SCOMPR(J2)
                LSTAR(I,1)=SLCC(OP,SCOMPR(J2))/100000
                ASTAR(I,1)=SAVGAV(OP,SCOMPR(J2))
                STGCNT(I)=1
            ENDIF
          ELSE
c ****
c * Does the same set of operations for the non-standby
c * assemblies
c ****
            M=0
            IF(OPTIND(OP) .EQ. 1) THEN
              OFLG(OP)=1
            ENDIF
            J10=0
```

```
              DO 860 J=1,NUMOP(R1)
                OP1=RQMTOP(R1,J)
c ****
c * Checks flag for dominated options
c ****
                IF(DFLG(OP1) .EQ. 0) THEN
                  J10=J10+1
                  COMPAR(J10)=OP1
                ENDIF
860           CONTINUE
              IF(J10 .GT. 1) THEN
              DO 862 J2=1,J10-1
                M=M+1
                OP2=COMPAR(J2)
                CM=COMPNM(OP2-NMOPTN)
                C1=0
                CALL CMPRTN(C1,CM,COMP)
                DO 864 J3=J2+1,J10
                  OP3=COMPAR(J3)
                  CM2=COMPNM(OP3-NMOPTN)
                  C2=0
                  CALL CMPRTN(C2,CM2,COMP)
                  T1=(LCC(C1)-LCC(C2))/100000
                  T2=GAMMA*(AVGAVL(C1)-AVGAVL(C2))
                  T3=T1/T2
c ****
c * Builds endpoint from comparison
c ****
                  IF(T3 .GT. 1.0) THEN
                    T3=1.0
                  ENDIF
c ****
c * Saves values of endpoint, L-star, A-star and choice
c ****
                  END(I,M)=T3
                  CHC(I,M)=OP3
                  LS(I,M)=LCC(C2)/100000
                  AS(I,M)=AVGAVL(C2)
                  M=M+1
                  END(I,M)=T3
                  CHC(I,M)=OP2
                  LS(I,M)=LCC(C1)/100000
                  AS(I,M)=AVGAVL(C1)
864             CONTINUE
862           CONTINUE
                  STGCNT(I)=M
                  CALL SRTEND(I,LS,AS,END,CHC,ENDPT,CHOICE,
     +                ASTAR,LSTAR,STGCNT,J10,K20)
                  STGCNT(I)=K20
```

```
c ****
c * Performs this routine only if one option dominated all
c * the other options
c ****
            ELSE
                CM=COMPNM(OP1-NMOPTN)
                C1=0
                CALL CMPRTN(C1,CM,COMP)
                ENDPT(I,1)=0.0
                CHOICE(I,1)=COMPAR(1)
                LSTAR(I,1)=LCC(C1)/100000
                ASTAR(I,1)=AVGAVL(C1)
              STGCNT(I)=1
            ENDIF
          ENDIF
        ENDIF
c ****
c * This set of routines is for the Case II stages that
c * were described in Ch. 4
c ****
        IF(LINK(R1,1) .EQ. '  0') GOTO 840
        IF(COMIND(R1) .EQ. 0) THEN
          LFLG=1
          SCNT=0
c ****
c * Counts the endpoints for all the options which branch
c * from this stage
c ****
          DO 870 J=1,NUMOP(R1)
            OP5=RQMTOP(R1,J)
            OPT(J)=OP5
            RQ2=OPRQLS(OP5,1)
            S1=0
            CALL STGRTN(S1,RQ2,STAGE)
            ST(J)=S1
            SCNT=SCNT+STGCNT(S1)
870       CONTINUE
          M=1
          LCNT=0
          K=1
          L=1
2045        T1=LSTAR(ST(1),K)-LSTAR(ST(2),L)
            T2=GAMMA*(ASTAR(ST(1),K)-ASTAR(ST(2),L))
            T3=T1/T2
c ****
c * Computes endpoint and checks to determine that the
c * value is not negative.  If it is, then one option
c * is dominated
c ****
            IF((T1 .LT. 0) .AND. (T2 .GT. 0)) THEN
```

```
c ****
c * Saves endpoint, L-star, A-star and option choice
c ****
               ENDPT(I,M)=ENDPT(ST(1),K)
               CHOICE(I,M)=OPT(1)
               ASTAR(I,M)=ASTAR(ST(1),K)
               LSTAR(I,M)=LSTAR(ST(1),K)
               IF(STGCNT(ST(1)) .EQ. 1) THEN
                 LCNT=SCNT-2
               ENDIF
               M=M+1
             ENDIF
c ****
c * Checks endpoint for negative value in the other
c * direction
c ****
             IF((T1 .GT. 0) .AND. (T2 .LT. 0)) THEN
               ENDPT(I,M)=ENDPT(ST(2),L)
               CHOICE(I,M)=OPT(2)
               ASTAR(I,M)=ASTAR(ST(2),L)
               LSTAR(I,M)=LSTAR(ST(2),L)
               M=M+1
             ENDIF
c ****
c * Performs this routine only if endpoint is non-negative
c ****
             IF(T3 .GT. 0) THEN
             IF(T3 .LT. 1) THEN
               IF(T3 .GT. ENDPT(ST(1),K)) THEN
                 ENDPT(I,M)=T3
               ELSE
                 ENDPT(I,M)=ENDPT(ST(1),K)
               ENDIF
c ****
c * Saves choice, A-star and L-star values, if option 1
c * is better
c ****
               CHOICE(I,M)=OPT(1)
               ASTAR(I,M)=ASTAR(ST(1),K)
               LSTAR(I,M)=LSTAR(ST(1),K)
               M=M+1
             ENDIF
             IF(T3 .GT. ENDPT(ST(2),L)) THEN
               ENDPT(I,M)=ENDPT(ST(2),L)
             ELSE
               ENDPT(I,M)=T3
             ENDIF
```

```
c ****
c * Saves choice, A-star and L-star values, if option 2 is
c * the better choice
c ****
            CHOICE(I,M)=OPT(2)
            ASTAR(I,M)=ASTAR(ST(2),L)
            LSTAR(I,M)=LSTAR(ST(2),L)
            M=M+1
            ENDIF
            IF(ENDPT(ST(1),K) .GT. ENDPT(ST(2),L)) THEN
              K=K+1
            ELSE
              L=L+1
            ENDIF
            STGCNT(I)=M-1
            LCNT=LCNT+1
          IF(LCNT .NE. SCNT-1) GOTO 2045
c ****
c * Counts number of endpoints computed from these rules
c ****
          DO 1350 L20=STGCNT(I),1,-1
            IF((ENDPT(I,L20) .EQ. 0) .AND.
               (L20 .LT. STGCNT(I))) THEN
               STGCNT(I) = L20
            ENDIF
1350      CONTINUE
        ENDIF
c ****
c * Begin Case III stages
c ****
        IF((LFLG .EQ. 1) .OR. (OFLG(OP) .EQ. 1)) THEN
          J7=0
          M=0
          IF(COMIND(R1) .EQ. 1) THEN
c ****
c * Performs these operations for non-standby stages
c ****
            IF(OPTIND(OP) .NE. 3) THEN
c ****
c * Gets the stage that the current stage is linked to
c ****
              RQ4=LINK(R1,1)
              S10=0
              CALL STGRTN(S10,RQ4,STAGE)
              DO 890 J5=1,NUMOP(R1)
                OP7=RQMTOP(R1,J5)
c ****
c * Checks to see if option was dominated
c ****
                IF(DFLG(OP7) .EQ. 0) THEN
```

```
                    J7=J7+1
                    OPT1(J7)=OP7
                    CM=COMPNM(OP7-NMOPTN)
                    C1=0
                    CALL CMPRTN(C1,CM,COMP)
                    PARE(J7)=C1
                  ENDIF
890             CONTINUE
c ****
c * Performs this set of operations only if there is more
c * than one option to compare
c ****
                IF (J7 .GT. 1) THEN
                  IF(AVGAVL(PARE(1)) .GT. AVGAVL(PARE(2))) THEN
                    K5=2
                    K6=1
                    STEP=-1
                  ELSE
                    K5=1
                    K6=2
                    STEP=1
                  ENDIF
                  L3=0
c ****
c * Builds on previous stage's endpoints
c ****
                  DO 892 J12=1,STGCNT(S10)-1
                    DO 894 J14=K5,K6,STEP
                      L3=L3+1
                      ATEMP(L3)=(ENDPT(S10,J12)/
     +                    AVGAVL(PARE(J14)))+0.00005
894                 CONTINUE
892               CONTINUE
                  L3=L3+1
                  ATEMP(L3)=0.0
c ****
c * Iterates for every endpoint
c ****
                  DO 896 L10=1,L3
c ****
c * Checks for negative endpoints
c ****
                    IF(ATEMP(L10) .LT. 1) THEN
                      TAS1=AVGAVL(PARE(1))*ATEMP(L10)
                      TAS2=AVGAVL(PARE(2))*ATEMP(L10)
                      DO 898 L12=STGCNT(S10),1,-1
                        IF(TAS1 .GE. ENDPT(S10,L12)) THEN
                          AST1=AVGAVL(PARE(1))*ASTAR(S10,L12)
                          LST1=LCC(PARE(1))/100000
                               +LSTAR(S10,L12)
```

```
                        ENDIF
                        IF(TAS2 .GE. ENDPT(S10,L12)) THEN
                          AST2=AVGAVL(PARE(2))*ASTAR(S10,L12)
                          LST2=LCC(PARE(2))/100000
                              +LSTAR(S10,L12)
                        ENDIF
898                   CONTINUE
                        T1=LST1-LST2
                        T2=GAMMA*(AST1-AST2)
                        T3=T1/T2
c ****
c * Computes new endpoint for comparison
c ****
                        M=M+1
                        IF(T3 .LT. ATEMP(L10)) THEN
                           END(I,M)=ATEMP(L10)
                        ELSE
                           END(I,M)=T3
                        ENDIF
c ****
c * Saves choice, A-star and L-star values
c ****
                        CHC(I,M)=OPT1(2)
                        AS(I,M)=AST2
                        LS(I,M)=LST2
                        M=M+1
                        END(I,M)=END(I,M-1)
                        CHC(I,M)=OPT1(1)
                        AS(I,M)=AST1
                        LS(I,M)=LST1
                      ENDIF
896                 CONTINUE
                    STGCNT(I)=M
c ****
c * Counts number of endpoints, calls routine to sort them
c ****
                    CALL SRTEND(I,LS,AS,END,CHC,ENDPT,CHOICE,
     +                         ASTAR,LSTAR,STGCNT,J7,K20)
                    STGCNT(I)=K20
                    ELSE
c ****
c * Performs this routine if only one option
c ****
                      DO 899 K7=1,STGCNT(S10)
                         BTEMP(K7)=ENDPT(S10,K7)/AVGAVL(PARE(1))
899                   CONTINUE
                      J15=0
```

```
                    DO 891 K7=1,STGCNT(S10)
                      IF(BTEMP(K7) .LT. 1) THEN
                        J15=J15+1
                        ENDPT(I,J15)=BTEMP(K7)
                        CHOICE(I,J15)=OPT1(1)
                        ASTAR(I,J15)=ASTAR(S10,K7)*
     +                                 AVGAVL(PARE(1))
                        LSTAR(I,J15)=LSTAR(S10,K7)+
     +                                 LCC(PARE(1))/100000
                      ENDIF
891                   CONTINUE
                      STGCNT(I)=J15
                    ENDIF
                  ELSE
c ****
c * Performs the same set of routines for the standby case
c ****
                    RQ4=LINK(R1,1)
                    S10=0
                    CALL STGRTN(S10,RQ4,STAGE)
                    J20=0
c ****
c * Checks the flag for dominated options
c ****
                    DO 1130 J18=1,CNT(OP)
                      IF(DSFLG(OP,J18) .EQ. 0) THEN
                        J20=J20+1
                        SCMPR(J20)=J18
                      ENDIF
1130                  CONTINUE
                      IF (J20 .GT. 1) THEN
                        IF(SAVGAV(OP,SCMPR(1)) .GT.
     +                              SAVGAV(OP,SCMPR(2))) THEN
                          K5=2
                          K6=1
                          STEP=-1
                        ELSE
                          K5=1
                          K6=2
                          STEP=1
                        ENDIF
                        L3=0
c ****
c * Builds new endpoints based on previous stage
c ****
                      DO 1132 J12=1,STGCNT(S10)-1
                        DO 1134 J14=K5,K6,STEP
                          L3=L3+1
                          ATEMP(L3)=(ENDPT(S10,J12)/
     +                        SAVGAV(OP,SCMPR(J14))) + 0.00005
```

```
1134                    CONTINUE
1132                  CONTINUE
                      L3=L3+1
                      ATEMP(L3)=0.0
                      DO 1136 L10=1,L3
                        IF(ATEMP(L10) .LT. 1) THEN
                          TAS1=SAVGAV(OP,SCMPR(1))*ATEMP(L10)
                          TAS2=SAVGAV(OP,SCMPR(2))*ATEMP(L10)
                       DO 1138 L12=STGCNT(S10),1,-1
                          IF(TAS1 .GE. ENDPT(S10,L12)) THEN
c ****
c * Rolls up new A-star and L-star values
c ****
                            AST1=SAVGAV(OP,SCMPR(1))*
     +                           ASTAR(S10,L12)
                            LST1=SLCC(OP,SCMPR(1))/100000
     +                           +LSTAR(S10,L12)
                          ENDIF
                          IF(TAS2 .GE. ENDPT(S10,L12)) THEN
                            AST2=SAVGAV(OP,SCMPR(2))*
     +                           ASTAR(S10,L12)
                            LST2=SLCC(OP,SCMPR(2))/100000
     +                           +LSTAR(S10,L12)
                          ENDIF
1138                    CONTINUE
                          T1=LST1-LST2
                          T2=GAMMA*(AST1-AST2)
                          T3=T1/T2
c ****
c * Computes new endpoints
c ****
                          M=M+1
                          IF(T3 .LT. ATEMP(L10)) THEN
                             END(I,M)=ATEMP(L10)
                          ELSE
                             END(I,M)=T3
                          ENDIF
c ****
c * Saves choice, L-star and A-star values
c ****
                          CHC(I,M)=SCMPR(2)
                          AS(I,M)=AST2
                          LS(I,M)=LST2
                          M=M+1
                          END(I,M)=END(I,M-1)
                          CHC(I,M)=SCMPR(1)
                          AS(I,M)=AST1
                          LS(I,M)=LST1
                      ENDIF
1136                CONTINUE
```

```
                    STGCNT(I)=M
                    CALL SRTEND(I,LS,AS,END,CHC,ENDPT,CHOICE,
     +                          ASTAR,LSTAR,STGCNT,J7,K20)
                    STGCNT(I)=K20
                    ELSE
c ****
c * Performs if only one option
c ****
                      DO 1399 K7=1,STGCNT(S10)
                         BTEMP(K7)=ENDPT(S10,K7)
     +                          /SAVGAV(OP,SCMPR(1))
1399                  CONTINUE
                      J15=0
                      DO 1391 K7=1,STGCNT(S10)
                        IF(BTEMP(K7) .LT. 1) THEN
                          J15=J15+1
                          ENDPT(I,J15)=BTEMP(K7)
                          CHOICE(I,J15)=SCMPR(1)
                          ASTAR(I,J15)=ASTAR(S10,K7)*
     +                           SAVGAV(OP,SCMPR(1))
                          LSTAR(I,J15)=LSTAR(S10,K7)+
     +                           SLCC(OP,SCMPR(1))/100000
                        ENDIF
1391                  CONTINUE
                      STGCNT(I)=J15
                    ENDIF
                  ENDIF
                ENDIF
              ENDIF
840     CONTINUE
        END

c ****
c * This subroutine is called by DECRUL, it sorts the endpoint
c * that are computed.  This makes comparisons easier at
c * later stages
c ****

        SUBROUTINE SRTEND(K,LS,AS,END,CHC,ENDPT,CHOICE,ASTAR,
       +           LSTAR,STGCNT,JCNT,L)

        INTEGER K,CHC(35,50),CHOICE(35,50),JCNT,STGCNT(25)
        INTEGER IHLD,CH
        REAL LS(35,50),AS(35,50),END(35,50),ENDPT(35,50)
        REAL LSTAR(35,50),ASTAR(35,50)
        REAL THLD,AHLD,CHLD
```

```
c ****
c * Performs a simple bubble sort and puts values into
c * descending order
c ****
      DO 1000 I=2,STGCNT(K)
        K1=I-1
        THLD=END(K,I)
        IHLD=CHC(K,I)
        AHLD=AS(K,I)
        CHLD=LS(K,I)
        DO 1002 J=1,K1
          J10=K1-J+1
          IF(THLD .GT. END(K,J10)) THEN
            TEMP=END(K,J10)
            END(K,J10)=THLD
            END(K,J10+1)=TEMP
            IEMP=CHC(K,J10)
            CHC(K,J10)=IHLD
            CHC(K,J10+1)=IEMP
            AEMP=AS(K,J10)
            AS(K,J10)=AHLD
            AS(K,J10+1)=AEMP
            CEMP=LS(K,J10)
            LS(K,J10)=CHLD
            LS(K,J10+1)=CEMP
          ENDIF
1002    CONTINUE
1000  CONTINUE

c ****
c * Once the choices have been sorted, they are compressed
c * into a smaller array
c ****

      L=0
      N=2
      IF(END(K,N) .LT. 1) THEN
        L=L+1
        ENDPT(K,L)=END(K,N)
        CHOICE(K,L)=CHC(K,N)
        ASTAR(K,L)=AS(K,N)
        LSTAR(K,L)=LS(K,N)
      ENDIF
      IF (JCNT .GT. 2) THEN
2047    N=N-1
        CH=CHOICE(K,N)
2049    N=N+1
        IF(CHC(K,N) .NE. CH) GOTO 2049
```

```
          IF(END(K,N) .LT. 1) THEN
            L=L+1
            ENDPT(K,L)=END(K,N)
            CHOICE(K,L)=CHC(K,N)
            ASTAR(K,L)=AS(K,N)
            LSTAR(K,L)=LS(K,N)
           ENDIF
        IF(L .NE. JCNT-1) GOTO 2047
        ENDIF
        L=L+1
        ENDPT(K,L)=0
        CHOICE(K,L)=CHC(K,N-1)
        ASTAR(K,L)=AS(K,N-1)
        LSTAR(K,L)=LS(K,N-1)
        END

c ****
c * This routine indexes the stages
c ****
        SUBROUTINE STGRTN(S1,RQ2,STAGE)
        CHARACTER*3 RQ2,STAGE(25)
        INTEGER S1
2051    S1=S1+1
        IF(STAGE(S1) .NE. RQ2) GOTO 2051
        END

c ****
c * This routine runs backward through the stages and plays
c * back the decision rules to make the choices
c ****
        SUBROUTINE PLAYBK(NR,STAGE,RQMT,COMIND,OPTIND,STGCNT,
       +            ENDPT,CHOICE,OPRQLS,LSTAR,ASTAR,OBACK,
       +            AVGAVL,SAVGAV,LINK,OPRQMT,NMOPTN,ENM,GAMMA,
       +            RPLC,PRDN,NUMLRU,EOQ,ROP,SEOQ,SROP,ORD,
       +            LOP,SORDLC,SROPLC,RN,COMPNM,COMP)

        INTEGER R1,S10,CHOOSE(20),COMIND(25),OPTIND(50)
        INTEGER CHOICE(35,50),OBACK(50),CHCSTN(10,50),GAMMA
        INTEGER OP1,ENM(5,10,25),STGCNT(25),OPRQMT(50)
        INTEGER PRDN(50),RN,C1
        REAL ENDPT(35,50),AVGAVL(50),SAVGAV(10,50)
        REAL LSTAR(35,50),ASTAR(35,50),OBFN(5000)
        REAL EOQ(50),ROP(50),SEOQ(10,25,50),SROP(10,25,50)
        REAL ORD(10,50),LOP(10,50),SORDLC(10,10,50)
        REAL SROPLC(10,10,50)
        CHARACTER*3 OPRQLS(50,25),RQ2,LINK(35,10)
        CHARACTER*3 STAGE(25),RQMT(25)
        CHARACTER*4 RPLC(50)
        CHARACTER*15 COMPNM(50),COMP(50),CM
        COMMON /OBJBLK/ OBFN
```

```
      DO 1120 I=1,20
        CHOOSE(I)=0
1120  CONTINUE

      I=NR
c ****
c * Begin with an availability of 1 for the last stage
c ****
      AV=1.0
2053    RQ2=STAGE(I)
        R1=0
        CALL RQR(R1,RQ2,RQMT)
c ****
c * Checks to see where the availability falls within
c * the computed endpoints and makes choice
c ****
        IF (COMIND(R1) .EQ. 0) THEN
c ****
c * Checks if stage is a component requirement
c ****
          DO 1100 J=STGCNT(I),1,-1
            IF(AV .GT. ENDPT(I,J)) THEN
              CHOOSE(I)=CHOICE(I,J)
            ENDIF
1100        CONTINUE
            RQ2=OPRQLS(CHOOSE(I),1)
          ELSE
c ****
c * Performs these operations if it is a component
c * requirement
c ****
            OP1=OBACK(R1)
            IF(OPTIND(OP1) .NE. 3) THEN
              DO 1102 K=STGCNT(I),1,-1
                IF(AV .GT. ENDPT(I,K)) THEN
                  CHOOSE(I)=CHOICE(I,K)
                ENDIF
1102            CONTINUE
                C1=0
                CM=COMPNM(CHOOSE(I)-NMOPTN)
                CALL CMPRTN(C1,CM,COMP)
c ****
c * Rolls up availability
c ****
              AV=AVGAVL(C1)*AV
              RQ2=LINK(R1,1)
            ELSE
c ****
c * Does this for non-standby assemblies
```

```
              DO 1104 L=STGCNT(I),1,-1
                IF(AV .GT. ENDPT(I,L)) THEN
                  CHOOSE(I)=CHOICE(I,L)
                ENDIF
1104          CONTINUE
              AV=SAVGAV(OP1,CHOOSE(I))*AV
              DO 1106 L2=1,OPRQMT(OP1)
                CHCSTN(OP1,L2)=ENM(OP1,CHOOSE(I),L2)
1106          CONTINUE
              RQ2=LINK(R1,1)
            ENDIF
          ENDIF
c ****
c * Loops until hits the final stage, this is determined
c * if the 'link' is equal to 0
c ****
          IF(RQ2 .NE. '  0') THEN
            S10=0
            CALL STGRTN(S10,RQ2,STAGE)
            I=S10
          ENDIF
        IF(RQ2 .NE. '  0') GOTO 2053
c ****
c * Writes configuration option choices to output file
c ****
        DO 1108 L5=1,NR
          RQ2=STAGE(L5)
          WRITE(20,*) 'FILL RQMT',RQ2,' WITH',CHOOSE(L5)
1108    CONTINUE

        DO 1110 L7=1,NMOPTN
          IF(OPTIND(L7) .EQ. 3) THEN
            DO 1112 L9=1,OPRQMT(L7)
              WRITE(20,*) 'FILL STNDBY',L7,L9,CHCSTN(L7,L9)
1112        CONTINUE
          ENDIF
1110    CONTINUE

        DO 1125 I=1,NUMLRU
          WRITE(20,*) 'REPAIR LOC FOR LRU ',I,' IS ',RPLC(I)
          WRITE(20,*) 'PROD CELL FOR LRU ',I,' IS ',PRDN(I)
1125    CONTINUE

        RN=RN+1
        WRITE(20,*) 'LCC OF PRODUCT IS,',LSTAR(NR,1)
        WRITE(20,*) 'AVAILABILITY OF PRODUCT IS,',ASTAR(NR,1)
        OBJ=GAMMA*(1-ASTAR(NR,1))+LSTAR(NR,1)
        OBFN(RN)=OBJ
        WRITE(20,*) 'MIN OBJ FUNCTION IS',OBFN(RN),'RUN ',RN
        END
```

Vita

Born in Bryan, Ohio on March 20, 1962, the author grew up mainly in the Midwest and graduated from Williamsville East High School, NY in 1979. She then attended Indiana University where she obtained a B.S. in Mathematics in 1983.

Upon graduation, the author was employed by the Air Force Logistics Command, at Wright-Patterson Air Force Base Ohio, to perform reliability analyses on Air Force weapon systems. In 1986, she went to work for Dynamics Research Corporation (DRC), a defense contractor, as an Operations Research Analyst, in the area of wartime planning and analysis. While working for DRC, she obtained an M.S. in Management Science from the University of Dayton in 1989.

She was then hired by Litton Computer Services as a Lead Engineer on the Reliability and Maintainability Information System (REMIS) project. She left the defense industry in 1990 to pursue a Ph.D. in Management Science at Virginia Tech, which she received in May 1994. During her time at Virginia Tech, she was employed as a graduate assistant and an Instructor of Management Science.

Melanie Hatch