COMPUTER SYSTEM SIMULATION OF AN ON-LINE INTERACTIVE COMMAND AND CONTROL SYSTEM

Herman Fischer

Litton Systems, Inc.

Data Systems Division

Van Nuys, Calif.

Abstract

A computer simulation model was used as an analysis "tool" for computer system design trade-offs for an on-line interactive command and control system preliminary design study project.

Three basic hardware configurations were modelled at the hardware interrupt/byte flow level:

a. A Centralized Dual Multiprocessor

b. Dual Computers

c. A Distributed System of Central and Remote Computers

The software of the system was modelled in several modules:

a. The Operating System Routines

b. The Data Base Management Routines

c. Interactive File Maintenance and Query Routines

d. Object-Coded Functional Applications Programs

e. Support and Control Software

Each module consists of parametric descriptions for each corresponding loadable software module (e.g., load module size, re-enterability) and procedural descriptions (e.g., read/write statements, processing statements, calls to system macros and other subroutines). The simulation is conducted in two modes: A fast-forward mode to load-up the system, and an observation mode to collect detailed data. A history tape of the environmental stimuli (interactive operator loads) on the system, produced by a separate "loading model", enters tasks into the system. The system model is implemented on the IBM 360, Model 65 Computer using a high level computer systems simulation language. The technique developed was so successful that it is now being actively used as a design tool and system performance evaluator, providing a means to minimize the technical risk in the development of all of Litton DSD's major software projects.

Study Description

This report documents the implementation methodology and results of a computer system simulation effort for a Preliminary Design Study (PDS), providing a definition of a selected candidate data processing and display system for a post-1975 military on-line interactive command and control system.

Program Overview

The PDS was a multi-tasked study oriented towards obtaining the design requirements for a post-1975 automated command and control center. The study effort culminated in a final report which forms a basis for actual system acquisition. The PDS was conducted in three basic tasks, over a period of eight months, to provide an opportunity for review and redirection at critical points of the system engineering process. Reviews were conducted by the responsible military agency.

A system model was built as part of the PDS in a successful and timely manner to evaluate alternative hardware trade-offs and to evaluate trade-offs in software design. The construction of the system model took about twelve man months of effort during the eight-month project. This included learning the modelling language, conceptualizing and designing the model, implementing, debugging, and validating the model, and conducting an intensive exercise of the model for trade-off analyses.

The major tasks of the PDS which related to the use of the model are now described.

A functional analysis task consisted of a thorough examination of the operational functions of the system with a definition of tasks and elements of information required to achieve those functions. Subsequently those tasks were allocated to man or machine.

A candidate system selection task was where the data processing and display (DP&D) system design alternatives were determined for trade-off study. The principal trade-off identified and recommended for analysis was the overall data processing system organization. The alternative configurations considered were:

a. A centralized multiprocessor

b. An organizationally oriented multicomputer

c. A distributed hierarachy of central and remote computers.

Since these configurations are considerably different in many aspects, the trade-off methodology employed in their analysis involved detailed simulation and performance evaluation.

The approach used in the trade-off analysis task reflected a classical trade-off methodology consisting of the following steps:

a. Definition of performance and design requirements.

b. Selection of alternatives capable of meeting those requirements.

c. Identification of a set of common parameters that can be evaluated and compared between alternatives.

d. Definition of measurement criteria that can be used to obtain a parametric evaluation of the alternatives.

e. Development of techniques and tools for evaluating the alternatives using the selected criteria.

f. Testing or evaluating each of the alternatives through use of the prescribed tools and techniques.

g. Selection of the alternatives that rank highest in the evaluation.

This framework provided the basis for conducting the trade-off analyses. However, for the system considered in the study, this framework was somewhat simplistic. There are far too many identifiable alternatives for a system of the complexity of this command and control system. Thus, with judicious application of design trade-offs and sound engineering judgement, it was possible to reduce the number of alternatives and resulting trade-offs to a number which could be effectively examined within the scope of the study.

A final step in the PDS was a formal system preliminary design, in which the results of the trade-off study were used to present a recommendation for the post-1975 command and control system. The use of the model provided the analysis detailed constraints on software design as well as validating a selection of a hardware configuration.

Modelling Objectives

The objective of the modelling and simulation tasks in the On-Line Interactive Command and Control System Preliminary Design Study was to provide a quantitative measurement of the performance of alternative system configurations performing the same functions. Since the system to be studied is essentially a time-sharing system responding to both internal interactive user activity and external message outputs, its performance cannot be stated simply in terms of throughput. Furthermore, the system configurations are diverse in nature but structured to give the same overall response time performance. Hence, it is inappropriate to assume that they all have the same relationship between response time and throughput. As a final complication, the system load is a mixture of a variety of tasks, each placing diverse requirements on the system. These tasks, many of which are similar in nature, are derived from a number of parallel operational functions of the system.

The combination of the above factors has led to the selection of a modeling and simulation technique for the determination of relative system performance of the various candidates. The objectives and use of modelling and simulation are:

a. To determine the load on the Data Processing and Display system (DP&D) in terms of generalized tasks or functions.

b. To determine the DP&D system configurations necessary to achieve the required performance under expected loads and conditions.

Modelling Tools Employed

The objectives specified above are achieved by the use of two distinct computer processed models. These models and their objectives, features, and use are shown in Table 1. A graphic representation of the model utilization is shown in Figure 1.

The specific use and relationships of the models are amplified in the subsequent sections.

The Load Generation Model

The Load Generation Model is used to determine the composite load on the DP&D system based on operational event occurrence rules derived from a functional analysis. The inputs to this program are card decks representing the event strings and relationships derived from the operational sequence diagrams generated in the Functional Analysis. The output of the Load Generation Model is a task occurrence history tape suitable for driving the System Model. Also obtained are statistical distributions of task occurrences useful in analysis and interpretation of the system load.

Characterizing the Data Processing Work Load

The DP&D system performance requirements are measured in terms of representative or generic tasks required to accomplish the command and control system functions. The system task load is specified as the occurrence rate or history of each generic task, and response time requirements are specified for certain groupings of these tasks. A generic task is a commonly occurring general purpose processing task which must be performed by the DP&D system in response to an interactive system user's request, an external event, or as part of an ongoing process. The functional analysis initially performed defined the generic tasks which must be performed by the DP&D system, and their method of employment.

The inputs to the Load Generation Model are sequences of either manual or data processing tasks required to accomplish the system functions. These task sequences are scheduled in time by the load generation model, so that the performance of these tasks will be representative of system operation. The output of the Load Generation Model is a task occurrence history which contains an ordered listing of the time occurrence of the generic tasks resulting from the functions modelled.

This history is stored on a disc file arbitrarily termed a "history tape." Each record of the file pertains to the occurrence of a single generic Data Processing and Display system task, defined by:

a. The time of occurrence of the task.

b. The generic task number.

c. The interactive system user by type and position, if an interactive user is involved in the task.

d. Any descriptive parameters necessary to further describe the task. (E.g., file number, number of records, or the type of file being accessed).

Table 1. Models in System Study Tasks

Item	Load Generation Model	Computer System Model
Objectives	To determine the load on the DP&D system in terms of generic tasks or functions.	To determine DP&D system configurations necessary to achieve required performance under expected loads.
Modelling Tools	Load Generation Model Uses a computer program developed in Fortran.	System Model A high level computer system simulation language.
Model Function	Models command and control center operations, determining the time relationship between manual and DP&D tasks of C&CS operational functions.	Models data processing and display systems: • Dual processor • Dual computer • Computer hierarchy
Model Operation	Driven by operational environment factors.	Driven by load generation model task history outputs.
Analysis	Utility programs provide statistical analysis of: • Interactive user utilization • Task profiles • Load profiles • Operational requirements	Standard statistics: • Utilization • Queueing • Trace Specialized statistics: • Task Response Times • Operational Response Times • Foreground/Background Throughput

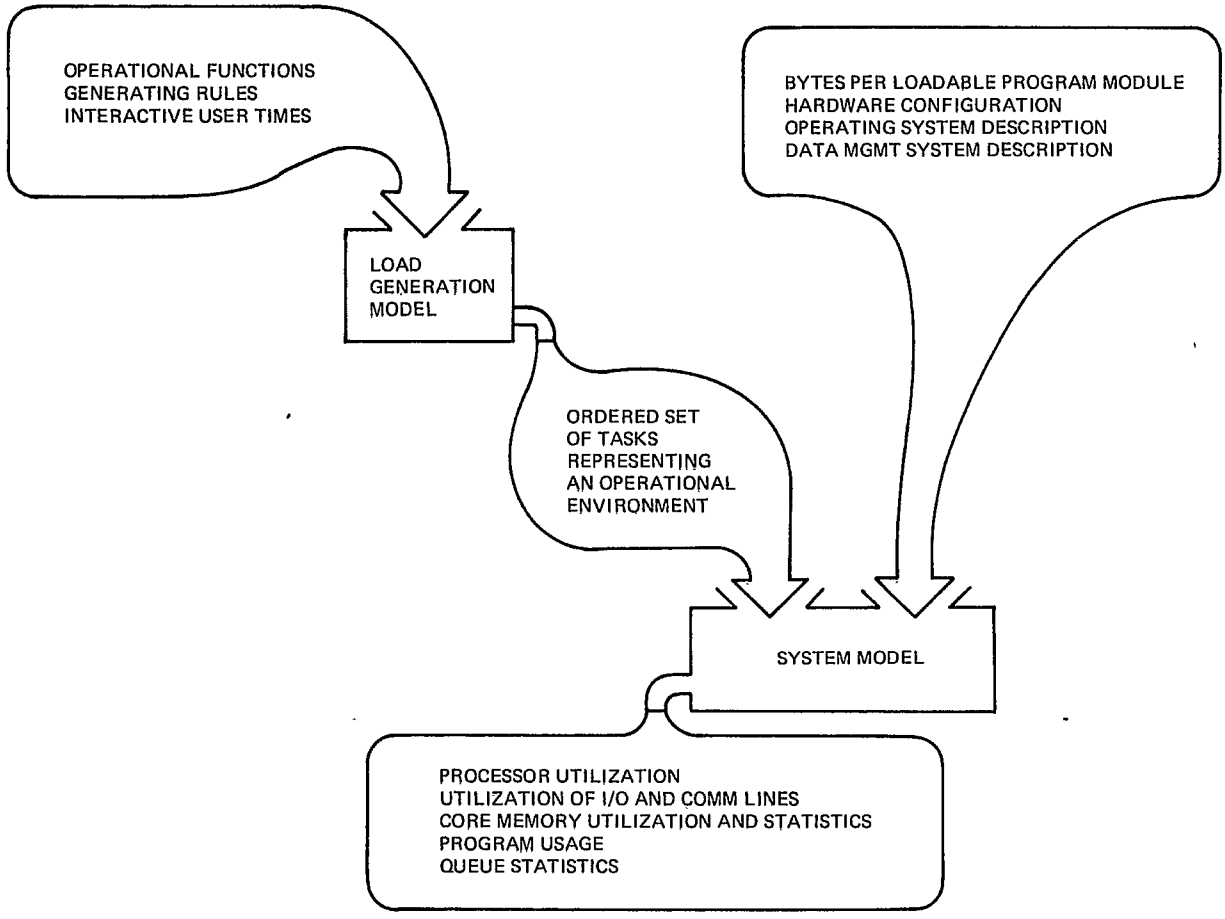

Figure 1. Model Utilization

The Load Generation Model also produces distribution functions for each of the generic task types. These distribution functions describe the inter-arrival rates of a given generic task as caused by all functions modelled, and the workloads imposed on all interactive system users.

Computer System Model

The Computer System Model is a computer system simulation of the three major computer configurations of the system trade-off study. As such, it is in reality three models, one for each principle system design. Since each system may also be configured for several varying load factors, variations on the three basic models have also been constructed.

The model itself is a large set of simulation language statements which describe both the hardware and software components which comprise the systems modeled. These descriptions are expressed in a format which is processed and interpreted by a high level computer system simulation language.

Modelling Language

The high level language was chosen for the modelling effort because of its capability to permit detailed examination of both the hardware and the software of the system. Hardware is simulated at the byte-flow/interrupt level. Each loadable software module is simulated by a parametric description and by a procedural description. The parametric description indicates module size, re-enterability, and task parameters. The procedural description consists of the procedural flow of read/write statements, CPU processing statements, and interfaces (calls) to system macros and other subroutines. This level of detail permitted trade-offs in software design to be accommodated for various hardware configurations. Thus it was possible to recommend not only an optimum hardware configuration for given operational requirements of the DP&D system, but also an optimum software design for each candidate hardware configuration.

Technique of Modelling

The basic objective of any simulation model is to create an entity which behaves to a realistic degree in the same manner as the system being simulated. Once this model has been created it is subjected to representative operational conditions or loads, and its performance is observed. If the model is a valid one, the performance of the modeled system, properly interpreted, is similar to the performance of the real system. The modelling process thus consists of three basic steps. First, the model itself is implemented and tested for proper operation. The second step, which is performed independently of the first step, is to determine the representative operational system loading conditions for which performance data is desired. The third step is to subject the model to the environmental loading conditions and to observe and interpret its performance.

Understanding the model thus consists of understanding these three steps. The major portion of this section is devoted to an explanation of the first step; that is, how does the model represent the system being evaluated. This section also describes how the model is subjected to its load conditions and how performance observations are made.

System Model Components

The System Simulation Model is composed of five basic components or sections. These are:

a. The Environmental Stimulator (or Control).

b. The Applications Programs (or Functional Software).

c. The Data Management System.

d. The Operating System.

e. The Hardware Configuration Descriptions.

Of the above components, the first pertains solely to the simulation process. Components b, c, and d correspond to or are models of computer programs which are intended for implementation in the eventual computer system. The last component is a model of the physical configuration of the eventual computer system.

Each of these components is constructed by a number of statements in the simulation language. The total complexity of the System Model is approximately 3000 statements for each of the three prime system alternatives. The majority of these statements, about 2300, are involved in the software components. The remaining 700 statements are almost entirely devoted to the control section, with the hardware description requiring less than 100 statements. Each of the above components are described in greater detail in the following subsections.

Environmental Stimulator

The Environmental Stimulator is that portion of the model which controls the simulation process and which stimulates the system to perform certain functions or tasks. The present model is relatively unusual in that a fixed externally specified sequence of events is used to drive the simulated system. This external event or task sequence is the actual workload of the system and is derived from appropriate operational scenarios by the Load Generation Model. The data processing workload is described in terms of history of task occurrences. The actual input to the Environmental Stimulator is a precise task sequence history rather than task occurrence rates. The use of such a sequence allows a high degree of repeatability which is valuable in establishing the models and in comparing alternative systems.

The Environmental Stimulator also controls the derivation and output of the data necessary to evaluate the performance of the simulated system. This output data consists of two types of information, a trace or event sequence listing, and summary statistics.

Since the expense of simulating extensive time periods is prohibitive, the Environmental Stimulator contains a control over the level of detail of the simulation process. A skimming mode, called fast forward, is implemented in the model and allows the simulator to proceed rapidly through the task sequences without gathering complete performance data. In this mode, approximations are made in the simulation of I/O operations in order to reduce simulation complexity. The normal mode, called observation mode, can then be used

selectively to snapshot regions of the task sequence which are of specific interest. The cost of running the model is about ten minutes of IBM 360/65 CPU time for approximately two hours of fast-forward simulated time and ten minutes of observation time.

Simulation of Hardware Configurations

The implementation of the hardware configurations in the System Model consists of statements which define the hardware entities which are modelled in the simulator. These hardware entities are described in terms of processing or throughput rate and the various paths which exist between them. The System Model also simulates the allocation and utilization of system resources which are limited, such as memory space. The manner in which the above hardware configuration features are specified is described briefly below. The number of the various types of entities which comprise the simulated system is specified to the model. The types of entities which correspond to physical items of equipment are:

a. Processors.

b. Multiplexer I/O units and controllers.

c. Paths to I/O devices.

d. Control Units.

Since the processors and multiplexers are the basic components of the system, their number is specified exactly. Usually the specified number of paths and control units are upper limits to the number of such entities expected.

A number of other types of entities are specified in the System Model which pertain primarily to the simulation process. Many reference tables are used as above to define specific configuration features and to accumulate statistical data.

Operating System

The operating system software of the System Simulation Model consists of a number of major programs which include a task initiator, an interrupt handler, and a program loader/scheduler. These routines are generally core-resident, although they may be made transient (disk/drum resident). In the multiprocessor and multicomputer configurations, there is one operating system. In the distributed hierarchy computer system the central computer, which manages the data base and performs applications tasks, has one operating system; and the peripheral computers, handling telecommunications and display processing, have their own operating system.

Data Management System

The two principle functions provided by the Data Management System are those relating to data base file maintenance and the interactive user query capability. These features are described below.

File Maintenance Capabilities. The file maintenance facilities may be called directly by an interactive user or by other generic tasks or programs. The file maintenance processor accepts fixed-format input from the operator to perform the following operations:

a. Add new data records to existing files.

b. Update existing records.

c. Delete existing records.

d. Purge unwanted or unneeded records from files on a periodic basis or when files are saturated.

The file-maintenance module is 'table-driven.' As an interactive user request is entered, a dictionary file provides tables showing the format for the interactive user input, the locations of each file data item on the display, the locations and forms of data items in a file record, validation criteria to be performed on certain data items when encountered, and security criteria for limiting access/updating of given file/data items. In addition, table-type specifications will show criteria for multiple-delete (purge) operations based on data contents of certain fields of file entries (e.g., expiration dates). The file maintenance module is located in a disk-resident program module. The simulated program length is approximately 8000 words.

Query Processing Module. The on-line Query Processing Module is presently located in a disk-resident program module. The simulated module size is approximately 7500 words, simulated by about 30 modelling language statements. The query module performs the following functions:

a. Accept structured query requests and establish data selection criteria to satisfy query.

b. Initiate retrieval of records from data-base files or other specified locations.

c. Determine whether these records do or do not meet interactive user-specified retrieval criteria.

d. Direct subsequent processing functions based on the evaluation of criteria comparisons (e.g., file maintenance, or further retrieval of hierarchical files).

Data Base Access

A set of macros are used to simulate the execution of data base access routines from application programs and/or generic task procedures. The macros load the required parameters for the data base access routines and provide linkage to the proper routine.

The database access routines, called by database access macros, allow a user to create, maintain, and retrieve data from multi-indexed (partially inverted) files which reside on the disk/drum. The size of the simulated access routines is 1500 words. The requestor is able to fetch records with embedded data meeting specific criteria. The specific functions which the data base access routines perform include the following:

a. Validate a request.

b. Maintain the status of every file within the system.

c. Retrieve data from the file.

d. Update the file.

The database access routines are currently core resident, due to the high demand for their services.

Simulation Outputs

The System Simulation Model provides a wealth of information about the system being simulated. This information is of two basic types:

a. Trace information is obtained during the execution of the model which describes the time sequence of events which occur.

b. Summary statistics data is accumulated by the model and printed at points specified by the user.

The trace information is derived from COMMENT statements which are inserted in the various simulated application and control programs. This allows complete user freedom in the required degree of visibility of the simulated system's behavior. Once the model is debugged, these trace data are typically limited to the start and completion of specific tasks or, for more complex tasks, the completion of significant task phases. Each comment which is printed during the trace output is annotated with its time of occurrence.

The summary data are printed in response to modelling language statements which are time-scheduled into the simulation process by the environmental control section. The summary data accumulated by the simulation processor are quite detailed. These data may also be augmented by user specified data which are accumulated in reference tables.

Much of the output data are provided for diagnostic purposes, and they are also generated when abnormal conditions arise. These type of data are principally current-status type of data. The data which are of interest in evaluating simulated system performance are typically one of the following types:

a. Utilization factors.

b. Resource utilization statistics.

c. Accumulation of occurrences.

d. Statistics on queues.

The specific items of information provided by the simulator which are of interest include:

a. Processor utilization – percentage of time that each CPU is busy.

b. Utilization and number of actions for each I/O channel, control unit, device and communication lines.

c. Core memory utilization, percentage in use, and minimum available.

d. Application program read and use counts.

e. Queue length, average and maximum, and time in queue for commands and tasks.

Instruction counts which are automatically tabulated by modelling statement line number are useful in assessing the number of times a specific simulated function has been performed. Complete generic task occurrence rates are obtained from a simulation run in this fashion. These rates differ from those tabulated by the Load Generation Model in that they include all executions of the functions rather than only these generated by external messages or interactive user activity.

In addition to the above data generated automatically by the simulator, there are a number of items developed by simulator control programs and made available through reference tables. These data include:

a. File access counts by file and type of access.

b. Total generic task elapsed time, occurrence count and mean elapsed time.

This latter datum is the single most important item produced by the system model, since it is a direct measure of response time for each generic task. These response times are used to derive the overall functional response times established as operational requirements.

While the current task status and instruction count data are normally regarded as diagnostic data, they are also useful in evaluating the performance of cyclic tasks and assessing the distribution of processing functions. The current values of iteration counts which are contained in task variables may be used to determine the number of times that a repetitive task has been executed. This count plus the iteration time allows a true time per iteration to be computed.

Use of Simulation Outputs in Trade Study

The output data provide statistics for the trade-off analyses. As has been mentioned, a common set of parameters was identified which can be evaluated and compared between alternative configurations. These parameters fall into the following categories:

a. Performance

b. Availability

c. Operational Suitability

d. General System Characteristics

e. Cost

A weighting function was formulated during the trade-off analysis to objectively evaluate the relative merits of each system on the basis of the common parameters. The benefit value assigned each parameter was derived by several means, including the use of system simulator model results. The following discussion will illustrate how a benefit value for system performance was derived; an example of how model results were used to establish a performance sensitivity for the multi-processor configuration; and an example of how CPU and equipment utilization figures were used to assign a benefit value to the three candidate configurations.

The performance characteristics were determined from those System Model outputs indicating the elapsed response time of the system to each generic task. In the case of interactive tasks, individual task response times were added together interspersed with interactive user (manual activity) times, to determine the overall man/machine system response to a given function. The functional response times were then compared with command and control system response requirements. The benefit value of a given system's performance was proportional to the difference between the system response requirement and actual observed simulated system response. A similar analysis occurred for background (noninteractive) functions.

Table 2 shows an evaluation of the performance of the standard multiprocessor configuration to show the sensitivity to four variations in configurations. The sensitivity is expressed in terms of percent of standard configuration throughout time for interactive and background tasks. For interactive tasks, throughput times shown are the sum of individual task throughput times, averaged in five minute intervals. For background tasks, the throughput times shown were the average per task throughput for a background task which occurred regularly throughout the peak activity period.

General system characteristics included considerations of CPU and I/O channel utilization statistics. Table 3 shows an example of the results of weighting the utilization statistics of the standard multiprocessor configuration with the standard multiple-computer and distributed-hiereichy-of-computers configurations. CPU and I/O channel utilization statistics shown were derived from system model runs in the peak 15 minutes of daily simulated system activity. A weighting formula, used for the tradeoff analysis task, assigned benefit points to utilization statistics as follows: Briefly, a utilization of 80 percent was assigned 100 benefit points, with 40 percent commanding a benefit point score of 200 points. I/O and CPU utilization are equally weighted. In each case, the worst case utilization is considered in establishing the benefit point scores. The results indicate the multiprocessor being superior to the multiple computer, and the distributed hierarchy of computers being approximately equal to the latter.

Conclusion

The modelling effort provided a means for obtaining a timely and comprehensive in-depth analysis of the requirements for an on-line interactive command and control system. The technique developed was so successful that it is now being applied to nearly all of Litton DSD's major software projects. In short, the experience gained from the modelling effort using a high level computer system simulation language as a tool for analysis was so useful that it has since been applied to help Litton DSD's software production projects spotlighting design problems and providing a method of minimizing the high technical risk normally associated with large software development projects.

Table 2. Example of Multiprocessor Performance Sensitivity

	Interactive Tasks		Background Tasks	
	Throughput Time	Percent Standard Configuration Time	Throughput Time	Percent Standard Configuration Time
1) Standard Configuration: 200 KIPS, 2 CPU 80 KW Memory Standard Load	166.2 sec	100%	45.1 sec/task	100%
2) Overload Condition: 200 KIPS, 2 CPU 80 KW Memory 200% Standard Load*	210.0 sec	126%	58.8 sec/task	130%
3) Degraded Mode: 200 KIPS, 1 CPU 80 KW Memory Standard Load	341.1 sec	205%	70.1 sec/task	155%
4) Extended Memory: 200 KIPS, 2 CPU 96 K Memory Standard Load	162.6 sec	98%	42.2 sec/task	94%
5) Increased CPU Speed: 300 KIP, 2 CPU 80K Memory 200% Standard Load*	204.6 sec	123%	34.0 sec/task	75%
*Interactive load only.				

Table 3. Example of CPU and I/O Channel Utilization

Item	Multiprocessor	Multiple Computer	Distributed Hierarchy of Computers
CPU Utilization	52% each processor	64% - Processor 1 3% - Processor 2	66% - Data base processor 3% - Each Distributed Processor
Benefit point score	170	140	135
I/O Channel Utilization	35% each processor	46% - Processor 1 2% - Processor 2	48% - Data base processor 2% - Each Distributed Processor
Benefit point score	212	185	180
Total Benefit point score	191	162	157