AISIM VAX VERSION TRAINING MANUAL

S. KNEEBURG
V. ALLERTON
T. WOODCOCK

Hughes Aircraft Co.
Ground Systems Group
P.O. Box 3310
Fullerton, CA 92634

February 1985

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

Prepared for

ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
DEPUTY FOR ACQUISITION LOGISTICS AND TECHNICAL OPERATIONS
HANSCOM AIR FORCE BASE, MASSACHUSETTS 01731
LEGAL NOTICE

When U.S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

OTHER NOTICES

Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

N. ANN KUO, 2LT, USAF
Project Manager,
Requirements Analysis

WILLIAM J. LETENDRE
Program Manager,
Computer Resource Management Technology

FOR THE COMMANDER

ROBERT J. KENT
Director, Computer Systems Engineering
Deputy for Acquisition Logistics
and Technical Operations
DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.
**REPORT DOCUMENTATION PAGE**

<table>
<thead>
<tr>
<th>1a. REPORT SECURITY CLASSIFICATION</th>
<th>1b. RESTRICTIVE MARKINGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNCLASSIFIED</td>
<td>NONE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2a. SECURITY CLASSIFICATION AUTHORITY</th>
<th>3. DISTRIBUTION/AVAILABILITY OF REPORT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. PERFORMING ORGANIZATION REPORT NUMBER(S)</th>
<th>5. MONITORING ORGANIZATION REPORT NUMBER(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESD-TR-85-129</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6a. NAME OF PERFORMING ORGANIZATION</th>
<th>6b. OFFICE SYMBOL (If applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HUGHES AIRCRAFT COMPANY</td>
<td></td>
</tr>
<tr>
<td>GROUND SYSTEMS GROUP</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7a. NAME OF MONITORING ORGANIZATION</th>
<th>7b. ADDRESS (City, State and ZIP Code)</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTER RESOURCE MANAGEMENT TECHNOLOGY</td>
<td>HANSCOM AFB, MA 01731</td>
</tr>
<tr>
<td>PROGRAM, DEPUTY FOR ACQUISITION (OVER)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8a. NAME OF FUNDING/SPONSORING ORGANIZATION</th>
<th>8b. OFFICE SYMBOL (If applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPUTER RESOURCE (OVER)</td>
<td>ESD/ALSE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER</th>
<th>10. SOURCE OF FUNDING NOS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>F33615-81-C-5098</td>
<td>PROGRAM ELEMENT NO. PROJECT NO. TASK NO. WORK UNIT NO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>11. TITLE (Include Security Classification)</th>
<th>12. PERSONAL AUTHOR(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AISIM VAX VERSION TRAINING MANUAL (U)</td>
<td>S. KNEEBURG, V. ALLERTON, T. WOODCOCK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13a. TYPE OF REPORT</th>
<th>13b. TIME COVERED</th>
<th>14. DATE OF REPORT (Yr., Mo., Day)</th>
<th>15. PAGE COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>FINAL</td>
<td></td>
<td>1985 FEBRUARY</td>
<td>154</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>16. SUPPLEMENTARY NOTATION</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>17. COSATI CODES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FIELD GROUP SUB.GR.</td>
<td>AISM DESIGN PROCESSES SIMULATION TRAINING MODELING ARCHITECTURE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AISM DESIGN PROCESSES SIMULATION TRAINING MODELING ARCHITECTURE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>19. ABSTRACT (Continue on reverse if necessary and identify by block number)</th>
</tr>
</thead>
</table>

THIS DOCUMENT IS THE TRAINING MANUAL FOR THE AUTOMATED INTERACTIVE SIMULATION MODELING SYSTEM (AISIM). THIS MANUAL PROVIDES STEP-BY-STEP INFORMATION NECESSARY TO BEGIN USING AISIM ON A VAX 11/780.
7a. NAME OF MONITORING ORGANIZATION (CONTINUED)
LOGISTICS AND TECHNICAL OPERATIONS

8a. NAME OF FUNDING/SPONSORING ORGANIZATION (CONTINUED)
MANAGEMENT TECHNOLOGY PROGRAM, DEPUTY FOR ACQUISITION LOGISTICS AND
TECHNICAL OPERATIONS
ACKNOWLEDGEMENTS

This manual was prepared by Hughes Aircraft Company under contract # F33615-81-C-5098. The work was sponsored by the Computer Engineering Applications Division (ALSE), Deputy for Acquisition Logistics and Technical Operations of the Electronic Systems Division (ESD) of the United States Air Force, Hanscom AFB, MA 01731. Funding for the effort was provided by the Air Force Computer Resource Management Technology Program, Program Element 64740F.

Program Element 64740F is the Air Force engineering development program established to develop and transfer into active use the technology, tools, and techniques needed to cope with the explosive growth in Air Force systems that use computer resources. The goals of the Program are to:

(a) Provide for the transition of computer system technology developments in laboratories, industry, and academia to Air Force systems;

(b) Develop and apply software acquisition management techniques to reduce life cycle costs;

(c) Provide improved software design tools;

(d) Address the various problems associated with computer security;

(e) Develop advanced software engineering tools, techniques, and systems;

(f) Support the implementation of high order languages, e.g. Ada;

(g) Address human engineering for computer systems; and

(h) Develop and apply computer simulation techniques for the acquisition process.
## CONTENTS

1. INTRODUCTION
   1.1 MODELING......................................................... 1
   1.2 OVERVIEW OF THE AISIM SYSTEM.................................. 1
   1.3 OVERVIEW OF AISIM MODELING CONSTRUCTS........................... 2
   1.3.1 ENTITIES REPRESENTING ELEMENTS EXTERNAL TO THE MODELED
         SYSTEM....................................................... 2
      1.3.1.1 The Load Entity........................................ 2
      1.3.1.2 The Scenario Entity.................................... 3
   1.3.2 ENTITIES REPRESENTING ELEMENTS INTERNAL TO THE MODELED
         SYSTEM....................................................... 3
      1.3.2.1 The Process Entity...................................... 3
      1.3.2.1.1 The Process Primitives................................. 3
      1.3.2.2 The Resource Entity.................................... 5
      1.3.2.3 The Action Entity..................................... 5
      1.3.2.4 The Legal Path Table.................................. 5
      1.3.2.5 The Queue Entity..................................... 5
   1.3.3 ENTITIES WHICH SUPPORT MATHEMATICAL OPERATIONS.............. 5
      1.3.3.1 The Constant Entity..................................... 5
      1.3.3.2 The Variable Entity..................................... 5
      1.3.3.3 The Table Entity....................................... 5

2. CREATING SYSTEM ARCHITECTURES
   2.1 DRAW AND NODRAW MODES........................................... 6
   2.2 DEFINING ATTRIBUTES FOR SYMBOLS................................ 8
   2.3 PLACING NODES ON THE GRID...................................... 9
   2.4 CONNECTING NODES............................................... 10
   2.5 CHANGING THE SIZE, TYPE, AND NAME OF NODES AND LINKS......... 12
   2.6 DELETING NODES AND LINKS..................................... 15
   2.7 MOVING PREVIOUSLY PLACED NODES................................ 15
   2.8 RECONNECTING EXISTING LINKS.................................. 16
   2.9 ALTERING ONE'S VIEW OF THE ARCHITECTURE GRID.................. 17
   2.10 DEFINING LEGAL PATHS......................................... 18

3. DEFINING PROCESSES IN THE DUI.................................. 23
   3.1 PROCESS EDITOR DRAW/NODRAW MODES............................... 25
   3.2 ACTION......................................................... 25
   3.3 HOLD.......................................................... 26
   3.4 ENTRY AND LOOP............................................... 28
   3.5 PROB, TEST, COMPARE AND BRANCH................................ 29
      3.5.1 PROB..................................................... 30
      3.5.2 TEST..................................................... 31
      3.5.3 COMPARE.................................................. 32
   3.6 VARIABLE MANIPULATION......................................... 34
   3.7 ITEM MANIPULATION............................................. 38
   3.8 RELATIONS AMONG PROCESSES.................................... 41
   3.9 RESOURCE ALLOCATION.......................................... 43
   3.10 CALL.......................................................... 47
4. REMAINING MODEL ELEMENTS................................................................. 49
  4.1 ACTIONS.......................................................................................... 49
  4.2 RESOURCES..................................................................................... 49
  4.3 QUEUES............................................................................................ 51
  4.4 CONSTANTS AND VARIABLES.......................................................... 51
  4.5 LOADS AND SCENARIOS................................................................. 52

5. A WORKING EXAMPLE............................................................................. 55
  5.1 DEFINING PROCESSES................................................................. 55
  5.2 REMAINING MODEL ELEMENTS...................................................... 59
    5.2.1 RESOURCE DEFINITIONS....................................................... 60
    5.2.2 QUEUE DEFINITIONS............................................................. 60
    5.2.3 ITEM DEFINITION................................................................. 60
    5.2.4 VARIABLE DEFINITION......................................................... 60
    5.2.5 ACTION DEFINITION.............................................................. 56
  5.3 LOADS AND SCENARIOS................................................................. 60

6. SIMULATION EXERCISES OF AISIM MODELS....................................... 62
  6.1 INITIALIZING A MODEL................................................................. 62
  6.2 DEFINING PLOTS.............................................................................. 63
  6.3 STARTING THE SIMULATION........................................................... 64
  6.4 EDITING VARIABLES BETWEEN SIMULATION STAGES..................... 64

7. A MORE ELABORATE EXAMPLE............................................................ 66
  7.1 MESSAGE ROUTING SUBMODEL...................................................... 66
  7.2 DEFINING ARCHITECTURAL ELEMENTS......................................... 66
  7.3 DEFINING REMAINING MODEL ELEMENTS...................................... 80
    7.3.1 RESOURCE DEFINITIONS.................................................... 80
    7.3.2 FILLING IN THE ACTION DEFINITIONS.................................. 81
    7.3.4 DEFINING LOADS AND SCENARIOS........................................ 81
  7.4 ANALYZING THE MODEL................................................................. 82

APPENDIX A - TERMINAL PROFILES FOR FORMS....................................... 83
APPENDIX B - SIMULATION REPORT FOR WORKING EXAMPLE....................... 86
APPENDIX C - SIMULATION REPORT FOR ELABORATE EXAMPLE.................... 99
<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>13</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
</tr>
<tr>
<td>10</td>
<td>14</td>
</tr>
<tr>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>14</td>
<td>19</td>
</tr>
<tr>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>21</td>
</tr>
<tr>
<td>17</td>
<td>22</td>
</tr>
<tr>
<td>18</td>
<td>24</td>
</tr>
<tr>
<td>19</td>
<td>24</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>21</td>
<td>26</td>
</tr>
<tr>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>23</td>
<td>28</td>
</tr>
<tr>
<td>24</td>
<td>29</td>
</tr>
<tr>
<td>25</td>
<td>29</td>
</tr>
<tr>
<td>26</td>
<td>30</td>
</tr>
<tr>
<td>27</td>
<td>30</td>
</tr>
<tr>
<td>28</td>
<td>31</td>
</tr>
<tr>
<td>29</td>
<td>31</td>
</tr>
<tr>
<td>30</td>
<td>32</td>
</tr>
<tr>
<td>31</td>
<td>33</td>
</tr>
<tr>
<td>32</td>
<td>34</td>
</tr>
<tr>
<td>33</td>
<td>35</td>
</tr>
<tr>
<td>34</td>
<td>35</td>
</tr>
<tr>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>36</td>
<td>37</td>
</tr>
<tr>
<td>37</td>
<td>38</td>
</tr>
<tr>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td>39</td>
<td>39</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>41</td>
<td>40</td>
</tr>
<tr>
<td>42</td>
<td>41</td>
</tr>
<tr>
<td>43</td>
<td>42</td>
</tr>
<tr>
<td>44</td>
<td>42</td>
</tr>
<tr>
<td>45</td>
<td>43</td>
</tr>
<tr>
<td>46</td>
<td>43</td>
</tr>
<tr>
<td>47</td>
<td>44</td>
</tr>
<tr>
<td>48</td>
<td>45</td>
</tr>
<tr>
<td>Page</td>
<td>Section</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>49</td>
<td>Forms for Primitives LOCK and UNLOCK</td>
</tr>
<tr>
<td>50</td>
<td>Process with Protected Resources</td>
</tr>
<tr>
<td>51</td>
<td>Secondary Form for Parameter-Passing Process</td>
</tr>
<tr>
<td>52</td>
<td>Form for CALL Primitive</td>
</tr>
<tr>
<td>53</td>
<td>Form for Action Entity</td>
</tr>
<tr>
<td>54</td>
<td>Form for Resource Entity</td>
</tr>
<tr>
<td>55</td>
<td>Form for Queue Entity</td>
</tr>
<tr>
<td>56</td>
<td>Forms for Constant and Variable</td>
</tr>
<tr>
<td>57</td>
<td>Form for Load Entity</td>
</tr>
<tr>
<td>58</td>
<td>Form for Scenario Entity</td>
</tr>
<tr>
<td>59</td>
<td>Transmitting Process</td>
</tr>
<tr>
<td>60</td>
<td>Receiving Process</td>
</tr>
<tr>
<td>61</td>
<td>Information Displayed on Entering the AUI</td>
</tr>
<tr>
<td>62</td>
<td>Aspects of Queue Behavior</td>
</tr>
<tr>
<td>63</td>
<td>Options for Statistics</td>
</tr>
<tr>
<td>64</td>
<td>Plot from Simple Example</td>
</tr>
<tr>
<td>65</td>
<td>System Architecture</td>
</tr>
<tr>
<td>66</td>
<td>Defined Resource Entities</td>
</tr>
<tr>
<td>67</td>
<td>Defined Action Entities</td>
</tr>
<tr>
<td>68</td>
<td>Defined Constant Entities</td>
</tr>
<tr>
<td>69</td>
<td>Terminal Profiles for Forms</td>
</tr>
<tr>
<td></td>
<td><strong>Total Pages</strong>: 84</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

This training manual presupposes virtually no programming experience and is intended to provide step by step information necessary to begin using AISIM as hosted on a VAX 11/780. It is not intended as a complete account of the system, and many topics covered in the companion AISIM User's Manual are covered here in less detail, or not at all. For further details on the operation of AISIM or on the kind of simulation AISIM is adapted to, the reader is referred to the more detailed AISIM User's Manual.

This manual consists of seven sections. This first section provides a brief overview of AISIM and its main concepts. Sections 2, 3 and 4 concern the Design User Interface (DUI), i.e., that part of AISIM in which models are created. Section 5 describes the complete construction of a simple model. Section 6 turns to the use of the Analysis User Interface—the part of AISIM where simulation and analysis occur—using the model developed in Section 5 as an example. Finally, Chapter 7 will document the creation, simulation and analysis of a more complex system.

1.1 MODELING

A computer model is a description of a system that is developed as a basis for calculations, predictions or further investigation. AISIM is especially designed to model systems that incorporate parallel processing. The purpose of an AISIM model is to give information on the workability of a system design, especially by providing statistics that serve to predict the operation of the modeled system if implemented.

Modeling is accomplished in AISIM by representing the elements of the system being modeled in terms of AISIM "entities." A detailed description of each entity is provided in Section 3 of the AISIM User's Manual. A general introduction to the types of system elements modeled by these AISIM entities is contained in section 1.3 of this manual.

1.2 OVERVIEW OF THE AISIM SYSTEM

AISIM consists of five subsystems, each of which performs a distinct function. These subsystems are: (1) the Design User Interface (DUI); (2) the Analysis User Interface (AUI); (3) the Replot User Interface (RUI); (4) the Hardcopy User Interface (HUI); and (5) the Library User Interface (LUI). Each of these subsystems is briefly described below.

(1) DESIGN USER INTERFACE

The DUI is the facility which enables the user to create or alter models of systems. It contains two sublevels, the Architecture Design Editor (ADE) and the Process Editor Interface (PEI). The ADE is used to model the physical layout of the given system, which is called the architecture. The PEI is used to define the processes or logic that are associated with that architecture. Other model entities are defined at the DUI level.
(2) ANALYSIS USER INTERFACE

With the AUI one subjects the model defined in the DUI function to simulation runs that test the behavior and response of the modeled system to various hypothetical conditions. In this function statistics are gathered on the operation of the system during simulation and, if desired, graphs of selected parameters are generated and are available for plotting.

(3) REPLYOT USER INTERFACE

The REPLYOT function enables the user to plot the statistics from various executions of a model, and to combine these plots as desired for future reference.

(4) HCOPY USER INTERFACE

The Hardcopy function provides the connection between the AISIM system and a printing device for automatically producing hardcopies of model logic. Process flowcharts constructed in the DUI can be printed on an HP2631G printer/plotter a TEK 4695 graphics copier, or the internal printer on the HP2623 terminal.

(5) LIBRARY USER INTERFACE

In the LUI the user is able to break apart and recombine parts of AISIM models, and obtain parts of models from a central system library. This feature is provided because some model components are used in other models and it is sometimes useful to store entire models for later reuse.

1.3 OVERVIEW OF AISIM MODELING CONSTRUCTS

This section provides a brief description of AISIM modeling constructs, to be followed by a more precise description of them in subsequent sections.

With some qualifications, AISIM's modeling constructs can be divided into the following four categories: (1) those used to represent the operations, properties, structure and internal relations of the modeled system itself; (2) those used to represent the environmental stimuli to which the system model is exposed; (3) those which represent the physical layout of the system; and (4) those which represent and facilitate mathematical operations.

1.3.1 ENTITIES REPRESENTING ELEMENTS EXTERNAL TO THE MODELED SYSTEM

1.3.1.1 The Load Entity. The Load entity is used to represent aspects of the modeled system that trigger processes within it. The Load entity represents the normal demand which is placed on the modeled system. Loads are defined by specifying the nodes at which certain Processes are to take place within a given period (see Scenario), together with specifications of several parameters which indicate the schedule that the Process triggering follows. The definition of a Load will also assign a priority to each of the Processes being triggered.
1.3.1.2 The Scenario Entity. A Scenario is used to represent the external demand on a system (i.e., Process triggerings from the outside) throughout a simulation exercise. The Scenario divides a simulation run into a number of periods that determine the frequency with which Loads will be initiated. They will also trigger Processes in a way that is not systematically related to the Loads in order to represent abnormal impositions on the system.

1.3.2 ENTITIES REPRESENTING ELEMENTS INTERNAL TO THE MODELED SYSTEM

1.3.2.1 The Process Entity. A Process is used to represent the operations, decisions, actions or activities that can be decomposed and defined in terms of more fundamental AISIM entities, called Primitives. A Process can take place in one or more of the system's nodes (or may execute independent of the nodes) and can make use of one or more Resources.

1.3.2.1.1 The Process Primitives. Primitives, of which there are 25, are the elements of which Processes are composed. A Process may be considered to be a collection of Primitives whose sequential execution describes the logic of the Process.

The 25 Primitives can be arranged into nine categories according to similarity of function. For the present, rather than give the meaning of each Primitive individually, it is sufficient to describe the categories and in a general way characterize the roles that members of each will play in the definition of a Process.

1. INTERNAL PROCESS EXECUTION CONTROL. The Primitives

   COMPARE
   BRANCH
   ENTRY
   PROB
   LOOP

serve as a "framework" for Processes, enabling the Process to branch (either unconditionally or under certain conditions) to another portion of the Process, or to repeat certain segments of the Process a specified number of times.

2. RESOURCE ALLOCATION. As mentioned earlier, a Process frequently competes with other Processes for Resources. The Primitives

   ALLOC
   DEALLOC
   RESET
   TEST
   LOCK
   UNLOCK

 govern the allocation of Resources among the various competing Processes.
3. PROCESS EXECUTION CONTROL. Since a principal feature of AISIM is its capacity to model parallel processing, i.e., distinct processes executing at the same time, these Primitives govern the timing of various processes in the system relative to one another. The Primitives

CALL
SEND
SUSPEND
RESUME
WAIT

will either interrupt the process in which they stand, or trigger or re-initiate some other process.

5. QUEUE HANDLING. The Primitives

FILE
FIND
REMOVE

govern the placement and retrieval of items in queues that have been defined by the user.

6. ITEM HANDLING. The Primitives

CREATE
DESTROY

govern the introduction and elimination of a system's transient data elements.

7. VARIABLE MANIPULATION. The Primitives

ASSIGN
EVAL

assign values to variables (both numerical and non-numerical) and allow for the mathematical manipulation of numerical variables.

8. TIME SEQUENCING. The Primitive

ACTION

which is associated with the Action entity described below, is included in process definitions to indicate the time a certain action (or process, decision, etc.) takes up without further describing the action's nature.

9. DEBUGGING. The Primitive

TRACE

is not used to represent a system's operations, but is rather provided as a debugging facility to aid the user in the task of tracing a history of process execution during simulations.
1.3.2.2 The Resource Entity. A Resource entity represents a component of the modeled system which may be necessary to the execution of a Process. Typically, a Resource will be required for more than one Process. Where several Processes demand a Resource that can serve only one Process at a time, all but one will stand in a queue until the Resource is available for them. The order in which the Processes will make use of the contended Resource is a function of the priorities associated with the various requests for the Resource.

1.3.2.3 The Action Entity. The Action entity is used in conjunction with the ACTION Primitive to represent any action, activity, decision, etc. that consumes time.

1.3.2.4 The Legal Path Table. The Legal Path Table (LPT) is a set of routes or paths between nodes in the system's architecture. The LPT is selected from all the possible paths between the nodes along the links, so that there is but one permissible routing of communication between the various nodes in the architecture. The LPT is accessed by several other elements of AISIM such as the EVAL Primitive, the keywords $NODE, $NXTNODE, and $LINK, and the Message Routing Submodel Processes.

1.3.2.5 The Queue Entity. A Queue represents any holding area, such as a memory buffer or job queue, for elements waiting to take up their role in the operation of the system. User-defined Queues can be used as a holding area for Items. A user-defined Queue can be manipulated in a number of ways described later and in the AISIM User's Manual section 3.4.

1.3.3 ENTITIES WHICH SUPPORT MATHEMATICAL OPERATIONS

1.3.3.1 The Constant Entity. A Constant is an entity whose value does not change during a simulation run. Constants are specified or altered in the DUI and can be edited before a simulation run in the AUI but cannot be changed (and do not change) once the execution of a model has begun. Several parameters required in the definition of an AISIM model, (e.g., the number of Resource units available, the period length of the simulation and the size for Queues) can only take Constants or simple numerics as values.

1.3.3.2 The Variable Entity. Variables, by contrast, are entities whose values can change during the exercise of a model. The majority of the parameters in the specification of a model can take Variables as values.

1.3.3.3 The Table Entity. Tables are single-value, single-argument functions defined by the user. They may be defined either as discrete, continuous, or alphanumeric and may have from 1 to 15 entries. Tables are accessed by the EVAL Primitive and serve as a supplement to the mathematical operations automatically available as part of the EVAL Primitive.
2. CREATING SYSTEM ARCHITECTURES

With the basic understanding of AISIM modeling concepts presented in the previous section, the reader should now be able to interact with the DUI. The exercises here are intended both to deepen the user's grasp of AISIM modeling constructs and to familiarize him with the prompts encountered while interacting with the DUI. In general, it is not a good idea to begin the design of an AISIM model without having done research and preparation on paper beforehand. However, as a teaching device, we shall develop fragments of an architecture from requirements formulated as we go along.

The method of logging on and invoking AISIM is computer-specific so we shall assume that the user has reached the point at which the computer prompts him with

AISIM READY

The user will have been offered a collection of information that looks something like that depicted in figure 1.

This is AISIM Production Version 4.0V which was built from AISIM Version 3.0V.
2/1/85

***Please report any problems to: Donald Constantine (617) 271-7754***

Figure 1. Typical Display Upon Entering the AISIM READY Level.

To enter the DUI, type

design project(test) term(trmtyp)

"test" is the name of the model to be designed. Trmtyp represents the type of terminal being used.

The valid terminal types are the following:

HP - HP2647A, HP2648A
HP23 - HP2623
VT - VT100
TEK - Tektronix 4105 with Selanar graphics

The user will be prompted with information that looks something like that shown in figure 2.
AISIM READY
design project(test) term(hp)
CURRENT PARAMETERS IN EFFECT:
VERSION: PRODUCTION VERSION 4.0
TERMINAL: HP
PROJECT: TEST
USER: [USER]
ENTER YES TO PROCEED, NO TO ABORT...

Figure 2. Typical Information on Entering the DUI

By typing

NO

the user will return to the AISIM READY level. Typing

YES

will put the user in the DUI and the screen will display an asterisk to indicate that the user may enter DUI commands.

When the computer displays the prompt "*", enter the Architecture Design Editor (ADE) by typing

ARCH

A grid like that in figure 3 will appear on the screen.

Figure 3. Grid on Which an Architecture is Designed
The AISIM constructs manipulated in the ADE are nodes which represent the hardware elements of a system and links which represent lines of communication between them. The physical layout of the system is represented by placing nodes and links on the grid to represent various hardware elements of a system and their (available) lines of communication. A Resource modeling entity is automatically created for each node or link when it is placed in the architecture.

As a mnemonic aid in distinguishing system elements, AISIM provides fourteen geometrical symbols for nodes. The symbols are called by the three-letter designations given in figure 4.

![Figure 4. Designations of the Fourteen Symbols](image)

With two exceptions these node symbols differ from one another only in their appearance. The two exceptions are the so-called "leaf-nodes" TTY and LOD. These nodes may be connected to the modeled architecture by only one link. All other nodes may be connected to any number of other nodes through any number of links. The rationale for this restriction is explained in the AISIM User's Manual, section 6.3.3.

2.1 DRAW AND NODRAW MODES

In the ADE there are two modes called DRAW mode and NODRAW mode. When the user is in DRAW mode, all architecture commands which change something in the architecture display cause the display to be updated immediately. If the user is in NODRAW mode, the architecture display is not automatically updated. The user can cause the display to be updated by typing
DRAW mode is the default on HP terminals and TEK4105 terminals. NODRAW mode is the only mode available on VT100 terminals. This restriction is due to a lack of capability in the terminal to make it operate like the other terminals. Therefore, if a user is on a VT100, he must type REDRAW to see the results of any ADE commands. The following discussions assume the user is on a terminal other than a VT100 except where specifically noted. If the user is on a VT100 terminal, he will need to use the REDRAW command to view the results of the ADE commands described in the following sections.

2.2 DEFINING ATTRIBUTES FOR SYMBOLS

As mentioned earlier, when a symbol is placed in the architecture, an AISIM entity called a Resource is created to represent the hardware element depicted by the node or link. Resources can have a number of user-named attributes. The DEFINE SYMBOL command allows the user to associate attributes with each symbol type so that when placed in the architecture, the Resource will be created with all attributes defined. For example, typing:

DEFINE SQR

(SQR could be replaced with any of the 14 symbol mnemonics or the mnemonic CON if a link is being defined.) The user will be prompted by a form as shown in figure 5.

```
RESOURCE NAME: 
TOTAL NUMBER OF UNITS: 
INITIAL NUMBER OF UNITS: 
DESCRIPTION: 
ATTRIBUTES

<table>
<thead>
<tr>
<th>NAME</th>
<th>VALUE</th>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 5. Symbol Definition Form
The user can add or modify fields within this form by using the terminal keys as defined in appendix A. Any values in the inverse video fields of the form are default values supplied from the AISIM design data base. The user may change these fields by typing over the existing values. The user may enter up to fifteen attribute names and related values of his choice into the attribute fields. For example, when defining attributes of a symbol type which is to represent a disk in the modeled system, the attribute names may be something like seek time, latency, etc., and the values would be the corresponding values for the particular disk being modeled.

A second use of the DEFINE SYMBOL command is the following:

```
DEFINE SYMBOL, RESOURCE NAME
```

where SYMBOL is one of the symbol mnemonics or CON, and RESOURCE NAME is the name of an existing Resource entity. If the named Resource entity exists, a form similar to the form shown above would be displayed. Instead of the default attributes, the form displayed would have the names and values of any attributes previously defined for the Resource entity referenced in the command.

This command will only be accepted if a Resource entity has been previously defined before entering the ADE. Since the user has not defined any Resource entities in his test data base yet, this command would fail. The user might want to try this command later.

2.3 PLACING NODES ON THE GRID

To place a node at a certain location on the grid--i.e., centered on that location--issue the PLACE command designating (1) the type of node to be placed, (2) a user-given name, and (3) horizontal and vertical position coordinates. One can also opt to indicate the size of the geometrical shape if the default value, equal to the number of characters in the user-given name, is unsuitable. To center a square named NODE1 twenty units from the left-hand side and thirty units from the bottom in figure 4 above, type

```
PLACE SQR, NODE1, 20, 30
```

Figure 6 shows the screen display that would result from this command.
All nodes are placed in this way. To place nodes in the positions shown in figure 7, type the following sequence of commands:

```
P TTY,NODE2,10,10
P PRP,NODE3,40,30
P TRI,NODE4,85,10
P TAP,NODE5,45,10
P CRD,NODE6,80,35
```
2.4 **CONNECTING NODES**

The second step in creating a system architecture is the placement of connections between the nodes. The CONNECT command is used to connect two nodes. Such connections, or "links", are defined by specifying (1) the node from which the link is to run, (2) the node to which the link is to run, and (3) a user-given name of the link. To place a link called "LINK1" from NODE1 to NODE2, type

```
CON NODE1,NODE2,LINK1
```

This command places a cursor at NODE1 if the user is on an HP terminal, or in the lower left corner if the user is on a TEK terminal; typing any alphanumeric character other than a period causes a straight line to be drawn between the centers of the two nodes, thereby drawing the link. The graphic result is shown in figure 8.

![Figure 7. Six Nodes on an Architectural Grid](image-url)
Figure 8. Architecture with One Link Defined

Links need not always appear as straight lines, as is shown in figure 9.

Figure 9. Architecture with a Bent Link
To create links that bend, enter the CON command as above. Then using the graphics cursor controls on the terminal (either the arrow keys on the HP terminals or the joystick button on the TEK terminal), the cursor can be moved to the spot where the link is to bend. When the cursor is at the point of the bend, type in a period (.). If no further bending is desired, typing any other non-period alphanumeric character will complete the connection. The resulting connection will resemble the one depicted above in figure 9.

Links may be given more than one bend by repeating the sequence of moving the cursor and typing a period (.), and then depressing any non-period character only when all the desired bends (up to 5 bends, i.e., six segments) have been created.

To create the links shown in Figure 10, type the following sequence of commands:

```
CON NODE1,NODE4,LINK4
CON NODE3,NODE6,LINK3
CON NODE3,NODE5,LINK5
```

NOTE: If the user is logged on through a VT100 terminal, there is no capability to create bent links. All connections are automatically made as straight lines between the two nodes.

![Architecture with Four Links](image)

Figure 10. Architecture with Four Links
2.5 CHANGING THE SIZE, TYPE, AND NAME OF NODES AND LINKS

The size, type, or name of nodes and the names of links can be changed using the CHANGE command. By typing the following commands, nodes and links may be altered:

CHG NAME, NODE1, NODEX
CHG TYPE, NODE2, LOD
CHG SIZE, NODE3, 7
CHG NAME, LINK4, LINKZ

The user may note the changes on his screen. By typing the following commands, the architecture is returned to its original configuration:

CHG NAME, NODEX, NODE1
CHG TYPE, NODE2, TTY
CHG SIZE, NODE3, 5
CHG NAME, LINKZ, LINK4

As mentioned earlier, Resource entities are created by the AISIM system to model the architecture elements. When the name or type of a node is changed or the name of a link is changed, the appropriate changes are also made to the associated Resource entities. That is, when a node or link name is changed, the associated Resource name is changed. When the type of a node is changed, new attributes may replace the existing attributes of the Resource since different attributes may be defined for the new symbol type. Refer to Section 2.2 of this manual.

2.6 DELETING NODES AND LINKS

Existing nodes and links may be deleted from a system architecture with the DELETE command. For this example, to eliminate the connection between NODE1 and NODE2 type

DELETE LINK1

The result on the screen would be that the link named "LINK1" would disappear.

When a node is deleted, all of the links associated with it also disappear. As an example type

DELETE NODE6

The result of deleting LINK1 and NODE6 is shown in figure 11. Note that LINK3 disappeared also.
2.7 MOVING PREVIOUSLY PLACED NODES

The location of a node on the architecture grid may be changed with the MOVE command. For example, to move NODE4 from its current position to the coordinates 55,5 one issues the command:

MOVE NODE4,55,5

The graphic result is shown in figure 12. The symbol is now centered at 55,5 with all of its previously defined connections intact.
2.8 RECONNECTING EXISTING LINKS

The previous example of moving NODE4 created a problem that can be solved with the command RECON. As figure 12 shows, the link between NODE1 and NODE4 now runs through NODE5. The connection can be made to bend around NODE5 by first typing

RECON LINK4

This command will delete the existing graphics for the link between NODE1 and NODE4 and, as in the original CONNECT command, place the cursor at NODE1. Using the sequence of cursor movements and periods (.) described in section 2.4, up to five bends in the existing connection between NODE1 and NODE4 can be created. To complete the connection, type any non-period character. The graphic result will be something like that shown in figure 13.
Figure 13. Architecture After Reconnecting

NOTE: The same restriction applies to the RECON command as the CONNECT command when the user is logged on to a VT100 terminal; i.e., only one-segment connections are allowed (see section 2.4).

2.9 ALTERING ONE'S VIEW OF THE ARCHITECTURE GRID

The usable grid space in ADE is actually larger than what can be displayed on the terminal screen at one time. If an architectural design is too large for the screen to accommodate, different parts of the total workspace can be viewed and manipulated through the WINDOW command. The WINDOW command allows the directional change of the user's view of the grid. The command specifies the direction of change---up, down, right or left---as well as the number of grid units the view is to be changed.

For example, to move the view of the screen in figure 13 down 15 units, type

WINDOW D,15

Figure 14 shows the result of this command.
The WINDOW command will accomplish both horizontal and vertical movements at the same time. To move our view of the screen further down 15 units and 20 units to the right, type

```
WINDOW D,15,R,15
```

Figure 15 shows the result of this command.
Note that the WINDOW command parameters required to get back to the original (HOME) position are always displayed above the upper right corner of the architecture grid.

2.10 DEFINING LEGAL PATHS

The purpose of the Architecture facility is to specify routes of communication between hardware elements so that process execution will be realistically related to the physical layout of a system. Such routes are represented by a Legal Path Table which specifies the links and the nodes through which communication from one node to another must take place. There are several methods of defining a Legal Path Table (LPT). Three methods are offered to the user at the end of an ADE session. These methods are predefined algorithms for the definition of an LPT which can be executed optionally at the user's discretion. See the AISIM User's Manual section 6.3.18 for details of how these algorithms function. For many architectures it is more economical to create the Legal Path Table while defining the configuration of hardware elements rather than using the algorithms mentioned above. If an LPT is generated according to the following discussion, the predefined algorithms should be bypassed since they would erase the LPT so defined.

Suppose we augment the architecture developed above with more links so that it resembles that shown in figure 16.
The LPT is defined by means of the command DEFINE PATH. If, for example, NODE1 is to communicate with NODE4 along the communication lines represented by the links LINK3, LINK5, and LINK2, type

```
DEFINE PATH,NODE1,NODE4,LINK3,LINK5,LINK2
```

No confirmation will be displayed immediately at the screen, but the Legal Path Table will have been augmented to reflect the new paths. However, the command LIST PATH enables the user to inspect the Legal Path definitions currently in effect. To obtain a listing at the screen of the legal path from NODE1 to NODE4, type

```
LIST PATH,NODE1,NODE4
```

The resulting list is shown in the upper right-hand corner of figure 17.
Figure 17. Typical List of Legal Paths Obtained in ADE

Note that paths from NODE3 and NODE5 to NODE4 have been automatically defined by the preceding DEFINE PATH command. The principle is that any time a legal path is defined through a number of nodes, the AISIM system creates legal paths from all nodes through which the path passes to the destination to node. Care should be taken in defining subsequent legal paths according to this method. Any conflicts of path routing in paths defined later would result in the elimination of previously defined paths. Following is an illustration of this operation of the system. Assume that the path has been created as above. If the user should now enter the command:

\[
\text{DEFINE PATH, NODE2, NODE4, LINK1, LINK4}
\]

not only would the path from NODE2 to NODE4 be established, but the path from NODE1 to NODE4 would be altered to be the direct path via LINK4. The paths defined automatically from the previous command (i.e., the paths from nodes NODE3 and NODE5 to NODE4) would still exist since there was no conflict with these paths and the newly defined path.
3. DEFINING PROCESSES IN THE DUI

Whereas the Architecture Design Editor (ADE) is used to represent the physical layout of a system, the Process Editor Interface (PEI) is used to represent the logic and data-handling behavior of processes in the system.

This section provides examples to familiarize the user with the commands and prompts used in the PEI. Earlier the user was urged to begin the design of an AISIM model with sufficient research and planning to fully understand the system to be modeled. However, as a teaching device, we shall develop fragments of a Process from requirements formulated as we go along.

The exercises here are intended both to deepen the user's grasp of the Process Primitives and to familiarize him with the prompts encountered while interacting with the PEI.

Assuming the user has just completed the foregoing section, entered an END command to exit the ADE sublevel, and another END command to bypass the LPT generation, he will be at the DUI level of operation. A "*' should be displayed. To invoke the PEI sublevel of the DUI, one enters the EDIT PROCESS command designating the name of the Process to be edited. Once in the PEI, the user can terminate the PEI session by entering an END command.

Consider first the simplest Process that could be of any use to the modeler of a system: the Process starts, a certain amount of time is taken with an Action and then the Process ends. This Process will be represented in AISIM by the START symbol, the ACTION Primitive and the END symbol. To represent such a Process in AISIM, one begins by issuing the command

```
EDIT PROCESS, EXAMPLE, NEW
```

which informs the system that one wishes to create a Process named "EXAMPLE". To alter a Process that has been previously defined, one would not enter the "NEW" part of the command. The computer will respond with a form to be filled in at the terminal. This is done by typing into the fields provided in the form. The form is shown in figure 18.
The NODE field asks for the node in the architecture with which the Process is associated. Since this Process is not yet related to an architecture, the field is left blank. The next field allows the assignment of attributes to the Process. For the present, we shall decline to do so. The field labeled PROCESS DESCRIPTION is for a comment to describe the Process. In this case, type "Example Process".

Depress the key that enters the form (see appendix A for specific key). The screen will go blank for a moment and then display the image depicted in figure 19.

Figure 19. Graphic Display That Follows Entering a Standard Process

Much of the information you entered on the form now appears in this graphic representation of EXAMPLE. The "NO" to the right of the START symbol indicates the Process has no attached attributes.
3.1 PROCESS EDITOR DRAW/NODRAW MODES

The Process editor maintains DRAW and NODRAW modes similar to the Architecture Design Editor except that both DRAW and NODRAW modes are available for all terminal types in the Process Editor. If the user is in DRAW mode, changes made to any Primitives on the screen will cause the specifically affected Primitives to be redrawn. If the user is in NODRAW mode, then any changes to Primitives will not cause the screen to be updated until a REDRAW command is entered, which will cause the current display to be redrawn. The user can also use the commands TOP, BOTTOM, UP and DOWN to cause the display to be redrawn at a different location.

The default mode for the Process Editor is DRAW mode. The user can switch to NODRAW mode or back to DRAW mode by entering the command NODRAW or DRAW. The following discussions assume the user in DRAW mode.

3.2 ACTION

To place an ACTION Primitive between the Start and the End symbols, enter the command

```
P ACTION
```

which tells the computer to place an ACTION Primitive between the last Primitive defined and the END symbol. The computer will now display a new form to be filled in. This is shown in figure 20.

```
PARAMETERS FOR ACTION

ACTION NAME: [field]
METHOD: [field]
MEAN TIME: [field]
DELTA-TIME: [field]
COMMENT: [field]
```

Figure 20. Form for the ACTION Primitive
The field ACTION NAME requests a name for the Primitive in the Process, which should be identical with the name of the associated Action entity (Action entities are described in section 4.1). We shall call it "Delay". The field COMMENT is a place to write a short reminder of what the ACTION Primitive is supposed to represent. The three remaining fields METHOD, MEAN TIME, and DELTA-TIME enable the user to vary the time taken up by the ACTION by invoking various statistical distribution methods (such as exponent, uniform, etc. See section 3.9.1 of the AISIM User's Manual for a description of the valid distributions.) Assume that the ACTION always requires the same amount of time, and hence the MEAN TIME requested will be equal to the duration of the ACTION. Indicate the time with a variable whose value for this example is specified elsewhere, calling it "T1".

The form should then be filled in thus: call the ACTION "Delay", set the method at "Constant", and set the MEAN TIME at "T1". Type the comment field "Action which causes delay". Leave the field labeled DELTA blank. After leaving the form, the screen will display a new version of EXAMPLE, as depicted in figure 21.

3.3 HOLD

EXAMPLE may be augmented in a number of ways. For example, the ACTION Delay may be repeated in succession. There are two ways to repeat this ACTION in a revised version of EXAMPLE. First, one can place more copies of Delay in the Process, one after another, with the command

\[ P \text{ ACTION} \]
This command may be repeated as many times as one wishes the Action to be performed in the Process. The second method of creating several instances of an ACTION which is less time-consuming, involves the HOLD storage area. HOLD constitutes a storage area for Primitives that are likely to be used more than once with little or no alteration. To place a previously defined Primitive into HOLD, type

\[
\text{HOLD 2}
\]

The number 2 represents the position in the Process of the Primitive to be stored in HOLD. The position is indicated by the numbers in the column on the left. Hereafter, the ACTION Primitive "Delay" may be placed in a Process by typing

\[
P \text{HOLD}
\]

Each time this latter command is issued, the user will be presented with the form associated with the Primitive in case any small alterations in its parameters are to be made. Whether or not any alterations are made, leaving the form will result in the placement of the Primitive stored in HOLD in the Process being edited.

Figure 22 shows the display that will appear after several identical ACTION Primitives have been placed in succession or after the HOLD storage area containing the ACTION "Delay" has been placed. The procedure of repetition will occur as many times as the user requests it.

![Example Process](image_url)

**Figure 22. Process with Three Identical ACTION Primitives**
3.4 ENTRY AND LOOP

If the ACTION "Delay" is to be performed a certain n number of times, as in the most recent version of EXAMPLE, a much simpler procedure is available than that of placing n instances of the ACTION between the START and END symbols. One can instead indicate more directly that a certain part of a Process is to repeat itself an n number of times. This is accomplished by means of the Primitives LOOP and ENTRY. Figure 23 shows EXAMPLE altered with LOOP and ENTRY Primitives to cause a triple repetition of the ACTION "Delay".

![Process Diagram](image)

Figure 23. Process with Triple Repetition of the ACTION Delay

The diamond-shaped figure indicates that the line of processing is to be diverted to the point labeled "Return" above it (it could have been given any label whatever up to 8 characters).

To effect the LOOP Primitive as shown in figure 23, we must first get rid of the two extra ACTION Primitives. This is done by typing the following commands:

```
DELETE 3
DELETE 3
P LOOP
```

The screen will show the form shown in figure 24.
PARAMETERS FOR LOOP:

LOOP TO LABEL: 

LOOP TIMES

COMMENT: 

Figure 24. Form for the LOOP Primitive

The first field asks for the name of the entry point to which Process control is to be diverted. The second asks for the number of times the primitives between the ENTRY and the LOOP are to be performed (i.e., control will be diverted to the entry point one less time than the number since the steps will have been executed once already when the LOOP is reached). The remaining field is self-explanatory.

The ENTRY Primitive must now be placed above the ACTION Primitive. Since the PLACE (or "P") command by default inserts the new Primitive just before the END symbol, a modified PLACE command is required to place a Primitive elsewhere in the sequence. To use this command, type

P ENTRY,2

The number "2" indicates where the Primitive is to be placed, in reference to the column of numbers on the left of the Process diagram.

The screen will then display the form shown in figure 25.

PARAMETERS FOR ENTRY:

ENTRY LABEL: 

COMMENT: 

Figure 25. Form for the ENTRY Primitive

The label will, in this case, be determined by the LOOP label previously defined. The ENTRY LABEL field should be entered exactly as in the LOOP LABEL, i.e., as "Return". Type an appropriate comment in the field provided, such as, "Entry From Loop Below". The result should be as in figure 23.

3.5 PROB, TEST, COMPARE AND BRANCH

Four other Primitives, PROB, COMPARE, BRANCH, and TEST are similar to LOOP in that they represent a branching to an ENTRY Primitive. EXAMPLE may be altered in the following four ways.
3.5.1 PROB. The PROB Primitive is used to indicate that the re-execution of the ACTION "Delay" has only a certain degree of probability.

Since AISIM has no command for directly replacing one Primitive with another, the existing Primitive LOOP must first be deleted.

Enter the command

```
DEL 4
```

where, as before, "4" indicates the location of the Primitive to be deleted. Figure 26 shows the display produced when the LOOP Primitive is deleted.

```
+----------------------------+
| START                      |
| 4.0                        |
+----------------------------+

```

RETURN ENTRY FROM LOOP BELOW

```
+----------------------------+
| DELAY                      |
+----------------------------+

```

ACTION WHICH CAUSES DELAY

```
+----------------------------+
| END                        |
+----------------------------+

```

Figure 26. EXAMPLE after Deletion of LOOP Primitive

The PLACE command is used to insert a PROB Primitive between the ACTION "Delay" and the END symbol. Type

```
P PROB
```

The screen will offer the form shown in figure 27.

```
PARAMETERS FOR PROBABILISTIC BRANCH:

BRANCH TO LABEL: [space]

PROBABILITY OF BRANCH: [space]

COMMENT: [space]
```

Figure 27. Form for PROB Primitive
Complete the first field with "Return". The second field should be filled in with the probability of branching, given as a percentage. Suppose there is a 25% chance of branching. Type the appropriate comment, "25% chance of branching". The resulting display diagram is given in figure 28.

Figure 28. Process with Probabilistic Branch

3.5.2 TEST. Another kind of branching Primitive is TEST. As mentioned earlier, Processes often make use of Resources for which there is competition. The TEST Primitive represents the procedure of ascertaining the availability of a given Resource and branching if that Resource is not available, or continuing if it is. This Primitive does not allocate the Resource. To place the TEST Primitive (after having deleted the PROB Primitive from the latest version of EXAMPLE), type

P TEST

The screen will display the form shown in figure 29.

PARAMETERS FOR TEST:

RESOURCE NAME: 
BRANCH TO LABEL IF NOT AVAILABLE
COMMENT: 

Figure 29. Form for TEST Primitive
In the RESOURCE NAME field, type the name of the Resource whose status is to be ascertained. The LABEL is the ENTRY Primitive to branch to, COMMENT is self-explanatory. If the PROB Primitive in the previous version of EXAMPLE is replaced by TEST (as in this example), EXAMPLE will now appear as in figure 30.

![Diagram of EXAMPLE PROCESS]

Figure 30. Process with TEST Branching

3.5.3 COMPARE In addition to probabilistic branching, AISIM also allows for conditional branching less specialized than the TEST Primitive. Most of these branchings will require the COMPARE Primitive. The COMPARE Primitive compares two numerical or alphanumerical values with respect to some relation and branches to a named ENTRY Primitive if the relation holds.

To place the COMPARE Primitive, delete the previously defined TEST Primitive and type

```
P COMPARE
```

The screen will display the form depicted in figure 31.
The fields OPERAND 1 and OPERAND 2 hold the parameters whose values are to be compared. The values may be represented by arbitrarily chosen names of variables (such as Var1 and Var2). They are compared with respect to the following six arithmetical relations indicated by the two letter code:

- EQ for "equal to"
- NE for "not equal to"
- GT for "greater than"
- LT for "less than"
- GE for "greater than or equal to"
- LE for "less than or equal to"

The BRANCH TO and COMMENT labels are now self-explanatory. The two QUALIFIER fields serve several purposes, the most important of which is to allow the comparison of attributes of entities as opposed to simple variables or numerics. The user should for the present disregard the complication posed by these fields and leave them empty. Fill in the OPERAND fields with arbitrarily chosen names of variables, "Var1" and "Var2". If the TEST Primitive is replaced by the COMPARE Primitive with the foregoing information entered on its form, the new version of EXAMPLE will be as displayed in figure 32.
And thus EXAMPLE is set to return control to the ENTRY Primitive if the value assigned to the variable Var1 is less than the value assigned to the variable Var2. These assignments are presumed to be made elsewhere.

3.6 VARIABLE MANIPULATION

In the previous example of the COMPARE Primitive, note that if the condition solicited is true, i.e., if VAR1 was less than Var2, EXAMPLE would perform the ACTION "Delay" indefinitely. On each occasion in which the comparison is made, the relation will hold and hence the Process will always be instructed to branch to Return. If neither variable changes its value, the Process will continue until it is halted by other causes (such as having a Resource necessary to it allocated elsewhere).

Using two new Primitives, ASSIGN and EVAL, we can alter EXAMPLE so that the ACTION "Delay" does not go on forever but only for a certain maximum time ("Maxtime"). This is accomplished with the ASSIGN Primitive which introduces a new variable for the accumulation of time consumed by the ACTION's execution times and by the EVAL Primitive, which recalculates the value of this accumulated time each time the ACTION is performed.

First, to command AISIM to place an ASSIGN Primitive between the START and the ENTRY, type,

```
P ASSIGN,2
```

The screen will now show the form displayed in figure 33.
For this example disregard the fields labeled Q1 and Q2; they serve the same purpose as do the QUALIFIER fields in the COMPARE Primitive. The purpose of this exercise is to create a temporarily useful, local variable, which we shall call "Acctime" whose value represents the amount of time that has been consumed in the repeated execution of "Delay". At the beginning of the Process the initial value of the variable will be zero. Hence, complete the V1 field with "Acctime" and the V2 field with "0". When this information is entered, the screen will display the graphic representation shown in figure 34.

To provide an apparatus for updating the variable "Acctime" on each occasion of the ACTION's execution, an EVAL Primitive must be placed
between the ACTION and COMPARE Primitives. To do this, type

```
P EVAL,5
```

The screen will display the form shown in figure 35.

![Figure 35. Form for EVAL Primitive]

The SET VARIABLE field holds the name of the variable whose value is to be calculated. The FUNCTION field contains the name of the operation to be performed on the two operands contained in the fields OPERAND1 and OPERAND2. A large variety of functional operations are available for this field (see AISIM User's Manual, section 3.9.11 for a list). For this example, the SET VARIABLE field should be "Acctime"; the Function, "Add"; OPERAND1, "TI" and OPERAND2 "Acctime". Type an appropriate comment, such as "Evaluating Acctime". The graphic representation of EXAMPLE will be as shown in figure 36.
This Primitive adds the current value of the variable "T1" to the value of "Acctime", producing an updated figure for the total time consumed by "Delay". Type an appropriate comment such as "Updating Accumulated Time".

The Process still requires alteration. The variables presently in the COMPARE Primitive must be changed from Var1 and Var2, respectively, to "Acctime" and "Maxtime". To do this, we must edit the COMPARE Primitive by typing

```
C 6 (cr)
```

This command tells AISIM that you wish to alter one or more of the previously defined parameters in the Primitive at location 6. The screen will display the form for the Primitive. It can be altered simply by writing over the existing information. When this is done and the form is "entered", EXAMPLE will satisfy the specifications for its alteration. Its graphic representation will be as in figure 37.
3.7 ITEM MANIPULATION

Another group of Primitives is categorized under the headings Queue Handling and Item Handling. These include CREATE, DESTROY, FILE, FIND and REMOVE. The Primitives CREATE and FILE will be used in this example. (Consult the AISIM User's Manual for information on the Primitives DESTROY, FIND and REMOVE.)

Consider the first version of EXAMPLE which consisted of the single ACTION Primitive "Delay". Suppose now that we conceive of EXAMPLE as a Process which gives rise to new data elements—messages, information, potential communications. This function of the Process may be represented by means of the CREATE Primitive, which represents the introduction of Items—the AISIM modeling entity that represents transient data elements—into the modeled system. To place the Primitive CREATE below the ACTION Delay in EXAMPLE type

P CREATE

Figure 37. Process with Comparative Branching
The form for CREATE is shown in figure 38.

PARAMETERS FOR CREATE

ITEMS TO BE CREATED ARE:

 COMMENT:

Figure 38. Form for CREATE Primitive

Complete this form with the names of the Items to be created. Enter the Item name "Msg" and an appropriate comment, "Transient Data Element". EXAMPLE will now appear as indicated in figure 39.

Figure 39. Item Creating Process

Items--transient data elements--can also be filed in holding areas called Queues with the FILE Primitive. To place a FILE Primitive below the CREATE Primitive in EXAMPLE, type

P FILE

The form for FILE is as shown in figure 40.
Complete the field FILE ITEM NAME with "Msg". The OPTION field tells where in the Queue the Item is to be placed. This location is specified relative either to absolute locations on the Queue ("FIRST" and "LAST") or relative to some other Item already on the Queue ("BEFORE" and "NEXT")*. The OPTION field will have as a default parameter LAST. In the ON QUEUE field enter "Msg-que". The graphic representation of this Process is indicated in figure 41.

*The method by which the system identifies the Item relative to which other Item is to be placed on a Queue (with the OPTION BEFORE or NEXT) need not concern us here. For more on this, see AISIM User's Manual, section 3.9.12.
3.8 RELATIONS AMONG PROCESSES

This section deals with the relationships that the execution of Processes bear to one another in an AISIM Model, and how one Process, and its execution, affects the execution of another. Processes affect one another's execution in four ways:

1. By sending Items that trigger the execution of another Process.
2. By triggering the execution of another Process through a CALL Primitive where the CALL may or may not pass parameters.
4. By resuming (with the RESUME Primitive) a Process which at some time suspended itself.

To understand how parameter and Item "passing" affect the execution of a Process, consider the form completed in the first version of EXAMPLE. In the form presented as a result of the command to edit a Process (i.e., E PROCESS,EXAMPLE,NEW), in the START field TYPE, the choices included "STD", "PARM" and "ITEM", standing for, respectively, "standard", "parameter-passing" and "Item-passing". These options are distinguished from one another in the following way. A Process can, before it is fully designed, be thought of as a "black box" whose internal workings are unknown. If the Process is conceived to be one that performs its function without having to be given anything in the way of information or data elements it will be a standard Process. If the Process requires certain data elements--discrete, countable entities--in order to execute, then it is an "Item-passing" Process. Finally, if the Process uses values of variables local to another Process, it is a parameter-passing Process.

For the first example, consider Item Passing Processes. In this exercise, delete the FILE and CREATE Primitives from EXAMPLE. To change EXAMPLE from a standard Process, as it now is, to an Item-passing Process, type

```
C 1
```

The screen will display the form originally filled out for EXAMPLE. Type "ITEM" over the existing "STD" in START field TYPE. Entering this, the screen will now show this secondary form on which Items needed by this Process are to be typed, as shown in figure 42.

```
ITEM PASSING START

ITEMS RECEIVED:

MUST ALL THE ITEM SERIAL NUMBERS MATCH (Y/N) [Y]
```

Figure 42. Secondary Form for Process
Type the single Item name "Msg" in the upper left field and type "N" in the match field. Entering this changes the Process representation so that it appears as in figure 43.

![Figure 43. Item Passing Process](image)

The figure to the left of the START figure indicates that the Process starts when, and only when, the Item MSG is delivered to it from some other Process.

None of the Primitives in the categories Item Handling and Queue Manipulation represents the delivery of Items to a Process. This delivery function is accomplished by the SEND. To exemplify SEND, a new Process, called "EXAMP-2", must be created. EXAMP-2 triggers the execution of EXAMPLE by delivering Items to it. For this example consider a Process identical except in name to the original EXAMPLE with the single ACTION Delay as depicted in figure 44.

![Figure 44. Process with ACTION Primitive](image)

Type the command

```
P SEND
```

The screen will display the form shown in figure 45.
Complete the SEND ITEMS TO field with "Example". In the first field of ITEMS TO BE SENT, type "Msg". Enter the comment "Sending Message Item". Figure 46 shows the graphic representation of the Process that will appear on the screen.

EXAMP-2 now triggers EXAMPLE by delivering to it Items required for its execution. The Item is automatically created by the SEND Primitive. An Item-passing Process may only be initiated through the SEND Primitive in some other Process, although the Items needed, and hence the Items sent, may be distributed among several Processes or several stages of a single Process.

3.9 RESOURCE ALLOCATION

As mentioned earlier, Processes in an AISIM model frequently make use of Resources. A Resource has a finite capacity which will limit the number of Processes it can accommodate at the same time. The five Primitives which relate to the allocation of such Resources are ALLOC, DEALLOC, RESET, LOCK and UNLOCK.
ALLOC and DEALLOC signal the allocation and deallocation of a Resource by the Process in which they appear. To place the ALLOC Primitive above the ACTION Primitive in EXAMPLE, type

\[ P \text{ALLOC,2} \]

To place a DEALLOC Primitive just above the SEND symbol in EXAMP-2, type

\[ P \text{DEALLOC,4} \]

The forms for these two Primitives are shown below in figure 47.

**Parameters for ALLOC:**
- **Allocate Resource Name:**
- **Number of Units Requested:**
- **Partial/All Allocation:**
- **Allocation Priority:**
- **Comment:**

**Parameters for DEALLOC:**
- **Deallocate Resource Name:**
- **Number of Units DEALLOCATED:**
- **Comment:**

Figure 47. Forms for Primitives ALLOC and DEALLOC

In each case, enter the name of the Resource to be allocated or deallocated, such as "Cpu", in the field provided. The field **NUMBER OF UNITS REQUESTED** holds the number of units of this Resource to be allocated or released. The **PARTIAL/ALL** field specifies the type of allocation scheme. Partial allocation will allocate Resources as they become available. All allocation means all of the units must be available at once. The **ALLOCATION PRIORITY** field specifies the priority at which Resources are allocated. Enter "1" for number of units and "$Priority" for priority. "$Priority" means to use the priority the Process was invoked with. Enter the appropriate comment, "Obtaining Cpu" or "Releasing Cpu" in the **COMMENT** field.

Placing these Primitives in EXAMP-2 (one above the ACTION and one below), produces a graphic representation like that shown in figure 48.
Allocating a Resource does not normally insure the uninterrupted availability of that Resource to a Process. Any Resources may be usurped by an ALLOC request with a higher priority. If the Process being modeled is one which, once begun, cannot be interrupted, the Primitives LOCK and UNLOCK must be used.

To obtain the forms for these Primitives,

\[
P \text{LOCK}, n
\]

or

\[
P \text{UNLOCK}, n
\]

where \( n \) is the position in the Process where the Primitive is to be placed. The forms for these Primitives are shown in figure 49.
These Primitives, if placed above the ALLOC Primitive and below the DEALLOC Primitive in EXAMP-2 would give a graphic representation like that shown in figure 50.

Figure 50. Process with Protected Resources
One final way to affect a Resource is through the RESET Primitive. It is used to reset the capacity of a Resource, where "capacity" is a measure of the number of Processes it will accommodate (support) at one time. For details on its use, see the AISIM User's Manual, section 3.9.18.

3.10 CALL

The function of the CALL Primitive is similar to that of the SEND Primitive, but whereas the SEND Primitive triggers Item-passing Processes, the CALL Primitive triggers both standard Processes and parameter-passing Processes. Thus, to understand how CALL works requires a brief discussion of parameter-passing Processes.

A parameter-passing Process is one that is "given" values for input variables and "returns" values for output variables. To create a parameter-passing Process, one would type "PARM" in the field START TYPE in the original form for Process. Entering this information on the Process form yields the secondary form shown in figure 51.

```
PARAMETERS FOR PASSING START

GIVEN:

RETURN:
```

Figure 51. Secondary Form for Parameter-Passing Process

On the form in figure 51, the user types the variables whose values are passed to the Process and the variables whose values are passed back.

The CALL Primitive values, i.e., parameters, are passed (given) to a called Process and returned to the calling Process. Parameter passing can occur only through the use of a CALL Primitive. A CALL Primitive is placed in a Process by typing

```
P CALL
```

The form for CALL is shown in figure 52.
PARAMETERS FOR CALL

CALLED-PROCESS NAME: 

WAIT/NOWAIT/BLOCK: 

GIVEN:

RETURNS:

COMMENT: 

Figure 52. Form for CALL Primitive

The field CALLED-PROCESS NAME asks for the name of the Process to be triggered. The field PRIORITY determines the priority associated with the called Process which may be used in cases of Resource contention. (It can be overridden by the priority specified in an ALLOC Primitive.) The GIVEN and RETURNS fields hold the local variables whose values are passed to and from the called Process. A CALL Primitive may trigger a standard Process and hence these fields may be empty. The COMMENT field is self-explanatory. The field labeled WAIT/NOWAIT/BLOCK determines whether the calling Process will wait for the called Process before continuing execution or will continue to execute independently of it. The reader is referred to the AISIM User's Manual for details on their use.
4. REMAINING MODEL ELEMENTS

Although the Processes and the Architecture are core modeling elements, their specification does not complete the task of model construction. They must be supplemented by definitions of other elements. These elements are grouped into two categories. The first category consists of those entities explicitly referred to in Processes, namely, Actions, Constants, (global) Variables, Tables, Queues and Resources. The second category consists of the two entities that are used to represent the impact of the environment on the modeled system. All these remaining entities are defined at the DUI level of AISIM operation.

The following two sections briefly describe the parameters, significance and principle commands associated with these remaining entities.

4.1 ACTIONS

Any ACTION Primitive placed in a Process must have a corresponding Action entity defined outside the Process. Such a definition is created by typing

E ACTION,ACTION NAME,NEW

The form for the Action entity is shown in figure 53.

ACTION: [action name]
CLASS: [optional class]
DESCRIPTION: [description]

Figure 53. Form For Action Entity

The ACTION field should hold the name of an Action referenced in some ACTION Primitive. The CLASS is an optional parameter for the user to provide a categorization--man, machine, etc.--of the sort of activity the Action represents. It functions as a second comment field. This field does not affect AISIM's operation and may be left blank. The field DESCRIPTION is for any convenient reminder of what the Action represents. It can be the same as the description of the corresponding ACTION Primitive.

4.2 RESOURCES

As mentioned earlier, any Resource mentioned in a Process--through the ALLOC, DEALLOC, or RESET Primitives--must be defined separately in the DUI. To create a new Resource, type

E RESOURCE,NAME,NEW
The screen will display the form shown in figure 54.

![Form For Resource Entity](image)

Figure 54. Form For Resource Entity

Complete the first field, RESOURCE NAME, with the name by which it is referred in any Process. The fields TOTAL NUMBER OF UNITS and INITIAL NUMBER OF UNITS indicate, respectively, the maximum number of Processes the Resource can accommodate at any one time and the number of Processes it can accommodate at the beginning of a simulation run (i.e., before being increased or decreased by the RESET Primitive). Enter the appropriate numbers. DESCRIPTION has its usual function. Type an appropriate description in the field provided. The fields under ATTRIBUTES indicate whether the Resource has associated with it attributes, including default attribute COST.

Up to fifteen attribute names may be entered with their initial values. The attribute COST functions as any other Resource attribute.

Though all Resources referred to require separate definitions, some Resources are defined automatically. For each node or link created in a model's network architecture, a Resource definition of the same name with default parameters is automatically written into the database. In other words, all nodes and links are identified with Resources. Thus, after an architecture has been created, the command

```
E RESOURCE, nodename
```

```
E RESOURCE, linkname
```

can be issued without having to indicate that the Resource entity is new (with "NEW"). Typically, however, not all of a system's Resources will
be represented in the architecture and not all of the Resources automatically created in ADE will have any positive role in the operation of the model. That is, such automatically defined Resources need not be invoked in the Process Primitives. Importantly, if an operative Resource is to be identified with an architectural element, it should be defined first in ADE and thereafter edited to provide it with suitable parameters (on the assumption that the default parameters are incorrect). ADE will not allow the definition of a node or link whose name is identical with that of a Resource already in existence.

4.3 QUEUES

Not all the Queues functioning in a system model need be defined by the user, since many are implicit in the operation of the system. The general rule is that any Queue manipulated by the FILE, FIND or REMOVE Primitives must be given a separate definition in the DUI, with the exception of a cross-reference set.

The cross-reference sets are explained in the AISIM User's Manual, section 3.5.2.

To define a new Queue, type

E QUEUE, NAME, NEW

The form for this entity is shown in figure 55.

QUEUE: [ ] SIZE: [ ]

DESCR: [ ]

Figure 55. Form for Queue Entity

The three fields should be filled in with, respectively, (a) the name of the Queue as found in the FILE, FIND, or REMOVE Primitive which uses the Queue, (b) the maximum number of Items that can be placed in it (the default value for which is "infinite") and (c) any useful reminder of the Queue's role in the modeled system.

4.4 CONSTANTS AND VARIABLES

Constants differ from global Variables only in that they do not change their values during the simulation exercise of a model. Once a value has been assigned to a Constant and a simulation is begun, its value is unchanging. Accidental attempts to alter the value of a Constant through the EVAL or ASSIGN Primitives will yeild an execution error.
message. The forms for Constants and Variables are quite similar and are called up by issuing the command EDIT CONSTANT or EDIT VARIABLE and then giving the name of the Constant or Variable.

The forms for Constants and Variables are shown in figure 56.

![Figure 56. Forms for Constant And Variable](image)

The fields CONSTANT and VARIABLE call for the entities' names. The VALUE fields call for numerical values for Constants and alpha-numerical values for Variables, and the DESCRIPTION fields call for any description.

4.5 **LOADS AND SCENARIOS**

The effect of the environment on a model is represented collectively by Loads and Scenarios. The relationship between Loads and Scenarios is this: Loads specify a number of Process triggerings to take place sometime during the simulation exercise of a model. Loads do not specify when the Process triggerings are to take place. They specify the distribution of the time passing between triggerings of the Process. Scenarios specify a collection of Loads and/or individual Processes together with a schedule indicating when the specified Loads or Processes are to be initiated.

To define a Load, type

```
E LOAD,NAME,NEW
```

The form for the Load Entity is shown in figure 57.
The LOAD field holds the name of the Load. The fields labeled NODE1 through NODE8 indicate the architectural nodes in which the Processes named in the Load take place. The field DESCR is for any helpful description.

The field labeled PROCESS holds up to five names of Processes. The fields SCHMDT, MEAN and DELTA together define the statistical method of distribution to be used in scheduling the Process triggerings. SCHMDT holds the name of the distribution method, MEAN holds the average time between Process initiations (in terms of the simulation clock) and DELTA is a second numerical parameter used to specify variation about the mean, if applicable. The field MAX # indicates the maximum number of Process instances to be initiated by this Load.

The Scenario entity defines the impact of the environment on the system for the entire simulation exercise of a model. In it the user specifies a number of "periods" into which a simulation exercise is to be divided, together with a uniform length each period is to have. The user then specifies a collection of Loads or Processes to be initiated at a specified time during the simulation. A priority is also given for each Process.

To define a Scenario, type

```
E SCENARIO,NAME,NEW
```

The form for the Scenario entity is shown in figure 58.
Figure 58. Form for Scenario Entity

The field SCENARIO holds the name of the entity. PERIOD LENGTH is the length of each period. The 14 fields labeled PERIODS are used to indicate the number of periods the Scenario is to have. The number of periods in the Scenario is determined by the number of these fields in which an entry is made. Any characters may be typed in these fields.

The fields labeled TRIGGER take the name of the Load or Process to be initiated. The field SCH TIME indicates the time at which the Load or Process named immediately to the left is to be initiated. The field PRIORITY is used to assign a priority to the named Process; it is ignored for a Load and should be left blank.
5. A WORKING EXAMPLE

This section documents the construction of an AISIM model that can be run through simulation tests and analyzed in the subsequent chapter. The model will be a representation of the transmitter/receiver relationship, an element of any communication system.

The transmitter/receiver relation modeled here is of the "polling" or "mailbox" type, as opposed to the "interrupt" type. In it, one transmitting Process generates messages and delivers them to a buffer. There the messages await treatment from a receiving Process. The transmitting and receiving Processes are not in direct communication with one another. Rather, the transmitter broadcasts messages according to need, and the receiving Process reads them from the buffer at intervals in accordance with expected need. In the system envisioned, transmission is randomized in two respects, (1) in the lengths of transmitted messages and (2) in the intervals between transmission. Reception is undertaken at regular intervals and the time consumed in processing a message is a linear function of its length.

The origination of a message in the transmitting Process will be represented by the creation of an Item (through the CREATE Primitive). The Item will have a variable attribute which will represent its length. Since the length will be randomized over a range of approximately 700 bytes, some mechanism must be incorporated for altering the variable attribute of each data Item (i.e., message). This is accomplished by (1) generating a random number in the range [0,1] subsequent to the creation of each Item, (2) multiplying the random number by twice the average message length and (3) assigning the number so obtained to the message length. This figure will then be used to calculate the time taken to send the message to the buffer (where it will be available to the receiving Process). Through an ACTION Primitive, the clock is then updated by the amount calculated.

In this system the buffer will not be manipulated by both the receiving and the transmitting Processes at the same time, so the buffer is considered a Resource and its allocation and deallocation by the ALLOC and DEALLOC Primitives will prevent it from being accessed simultaneously by both Processes.

5.1 DEFINING PROCESSES

This description of the transmitting Process gives the steps of its execution. The transmitting Process:

(1) Starts
(2) Allocates one unit of a Resource BUF1 representing the buffer
(3) Creates a message, represented as an Item called "Msg"
(4) Generates a random number between 0 and 1
(5) Multiplies the random number generated by twice the average message length

(6) Assigns the number obtained in the previous step to the Item attribute representing the message length

(7) Calculates the delay time which is proportional to the message length (i.e., an amount equal to the message length divided by the transmission rate in seconds per byte)

(8) Delay for the calculated amount of time

(9) Delivers the message item to the Queue called Buffer through the FILE Primitive

(10) Releases the Resource BUF1 representing the buffer

Figure 59 shows the Process flowchart derived from this description.
The receiving process will first determine whether or not the buffer is being manipulated by the other process by testing for utilization of the Resource call BUFI. If the Resource is in use, the process will abort by branching to the END symbol. If BUFI is free, the process will read...
the next message from the buffer, and calculate a receiving time in roughly the same way that the transmitting time for that same Item was determined in the transmitting Process. The clock is then updated by the amount of time calculated.

This description can be expanded into more specific design requirements. The receiving Process

(1) Starts

(2) Tests for the availability of the buffer by determining whether or not the Resource is in use through the TEST Primitive. If so, the Processes execution will branch to the END symbol.

(3) The next message Item on the Queue called Buffer is read via the REMOVE Primitive.

(4) If there is nothing on the buffer, Process execution, as in step (2), branches to the END. This step is represented by a COMPARE Primitive.

(5) The message length is assigned to a local variable through the ASSIGN Primitive.

(6) A receiving time is calculated to be proportional to the message length (i.e., equal to the message length times some reception speed in seconds per byte).

(7) The clock is updated through the ACTION Primitive in the amount required to receive the message.

(8) The message Item, having been read, is eliminated from the system through the DESTROY Primitive.

(9) An ENTRY Primitive is inserted just before the END symbol of the Process to indicate where execution is to resume from the branchings in steps (2) and (4).

Figure 60 shows the flowchart representation of the Process derived from these requirements.
5.2 REMAINING MODEL ELEMENTS

The remaining model entities must now be defined. These include all the entities mentioned in any Process Primitive. These are the following:

1) The Queue named "Buffer" onto which messages are filed;

2) The Resource Buf1 which represents a device to protect the buffer against manipulation by two Processes at once;

3) The Item Msg, each instance of which is to represent a message transmitted onto the Queue;
4) The global Variables named Gammal, which is twice the average message length; and Gamma2, which is the transmission rate.

5) The Action entities named Read-msg, which charges time for reading a message, and Sending, which charges time for sending a message.

5.2.1 RESOURCE DEFINITIONS

The Resource BUF1 is given proper parameters. It will have only one initial unit and will have a maximum of one. It will retain the default of no attributes and a cost of zero. An appropriate description is: "Resource Associated With Buffer".

5.2.2 QUEUE DEFINITIONS

The Queue called "Buffer", which is accessed by the FILE and REMOVE Primitives, will retain its default size of "Infinite". A helpful description is: "Buffer On Which Messages Are Stored".

5.2.3 ITEM DEFINITION

The Item Msg which represents messages transmitted and received will have one attribute called "Length". Its initial value will be the literal "$Length", since the value of this attribute will always be assigned within the Process that transmits it to the buffer.

5.2.4 VARIABLE DEFINITION

The Variables "Gammal" and "Gamma2" are defined with initial values of .700 and .002 respectively. These values are used in calculating the transmission and reception time. Gammal is twice the average message length, and Gamma2 is the transmission rate.

5.2.5 ACTION DEFINITION

Action entities "Sending" and "Read-Msg" must be defined in order to satisfy the references in the ACTION Primitives in the Processes Transmit and Receive. The class and description fields can be filled in as desired by the user. These fields have no effect on the simulation.

5.3 LOADS AND SCENARIOS

Finally, we must define the hypothetical conditions to which the modeled system will be exposed. Six Loads are defined for this model. L1, L2 and L3 each trigger the transmitting Process. L11, L22 and L33 each trigger the receiving Process in a schedule of expected need associated with L1, L2 and L3. The triggerings of the transmitting Process are randomized, whereas the triggerings of the receiving Process are scheduled at regular intervals. The complete Load definitions are found on pages 3 through 5 of the Model Verification Report which appears in Appendix B.
The Scenario for this model consists of six periods. The Loads are distributed throughout the simulation period as follows: each pair of Loads is triggered at intervals of 200 units on the simulation clock. The complete definition is found on page 5 of the Model Verification Report in Appendix B.
6. SIMULATION EXERCISES OF AISIM MODELS

The model is now ready to be run through a simulation exercise to determine its behavior under the defined environmental conditions. To begin this exercise, enter the Analysis User Interface (AUI) from the AISIM READY level by typing

A P(projectname)

Projectname is the name of the model we wish to expose to a simulation exercise. The user will be prompted with information that will look something like that shown in figure 61.

CURRENT PARAMETERS IN EFFECT:
VERSION: PRODUCTION VERSION 4.0
TERMINAL: HP
PROJECT: TEST
USER: [USER]
ENTER YES TO PROCEED, NO TO ABORT...

Figure 61. Information Displayed on Entering The AUI

After declining the abort prompt by typing

YES

and following the translation of the model, the user is in a position to issue commands before the execution of the simulation.

6.1 INITIALIZING A MODEL

If more than one Scenario has been defined, the system will ask

WHICH SCENARIO DO YOU WISH TO TRANSLATE?

Type the name of the Scenario that defines the environmental conditions to which the model is to be subjected. For this model, we have defined only one Scenario so the program will perform model initialization. If no errors are detected at this stage the computer will prompt with,

NO ERRORS DETECTED DURING MODEL TRANSLATION
YOU MAY NOW ENTER COMMANDS

If an error had been made, AISIM would have prompted with

ERRORS DETECTED IN MODEL TRANSLATION

This prompt indicates that some aspect of the model definition is in error. If such is the case, determine where the errors are, see Appendix B of the AISIM User's Manual for a description of the error messages, and return to the DUI to correct them. The matter of getting to the DUI has already been covered in previous chapters. For this example, assume that the AISIM model is properly defined.
6.2 DEFINING PLOTS

Two choices are available at this point: Proceed to the simulation exercise of the model or request that graphs of some of the activities monitored during the simulation be defined so that they can later be inspected at the terminal.

For example, in the model under consideration, one of our main concerns is to determine whether the buffer onto which the transmitting Process places messages (and from which the receiving Process retrieves the messages) reaches some maximum burden or whether it shows a tendency to infinite queueing. To produce a graph of the behavior of the buffer we type

```
DEF QUEUE,BUFFER
```

The screen will display a selection of the aspects of the behavior of a Queue for which a graph can be defined. These are shown in figure 62.

```
ATTRIBUTES (PLACE AN X NEXT TO ONLY ONE)

NUMBER IN QUEUE
NUMBER BLOCKED
TIME IN QUEUE
TIME BLOCKED
```

Figure 62. Aspects of Queue Behavior

To define a graph showing the number of Items in the Buffer, we would enter an "x" for "NUMBER IN QUEUE". The screen would then display the options for defining the type statistic on the number of Items in the Queue. These options are shown in figure 63.

```
STATISTICS (PLACE AN X NEXT TO ONLY ONE)

CURRENT
CUMULATIVE MESH
CUM STANDARD DEV
CUMULATIVE MIN
CUMULATIVE MAX
PERIOD MEAN
PER STANDARD DEV
PERIOD MIN
PERIOD MAX
```

Figure 63. Options for Statistics

To calculate the current number of message Items in the Queue called "Buffer" at any given time, enter an "X" next to "CURRENT". The entities with respect to which graphs can be defined are Resources, Queues, Processes, Items and Variables. Up to ten such graphs may be defined per analysis session.
6.3 STARTING THE SIMULATION

Once the model is initialized and graphs are defined, the model may be executed through a simulation run. The execution of the model may be triggered either for the entire Scenario or for a specified number of periods, so that global Variables can be given new values different from those previously defined in the DUI.

The values of Constants and the initial values of global Variables may also be changed before a simulation exercise begins. The latter option will be chosen in this example to investigate the effect of altering the time required to transmit or to process message Items. To begin the simulation, type

GO 1

This command indicates that the simulation is to be run for 1 of the simulation periods defined when the model was created in the DUI. When this first stage of the simulation is completed the screen will offer the following message:

END OF PERIOD
YOU MAY NOW ENTER COMMANDS

6.4 EDITING VARIABLES BETWEEN SIMULATION STAGES

To change the value of a variable, issue the appropriate command with information as to (a) the type of entity to be edited, (b) the name of the entity whose value is to be changed and (c) the new numerical value of the entity. The Variable Gamma2 formerly had the value of .002. To change it to .001, type

E V, gamma2, .001

The simulation may be continued for two more periods by typing

GO 2

When this stage of the simulation is completed, the value of Variables may be changed back to .002 by typing

E V, gamma2, .002

To command that the remainder of the Scenario be run through without further interruption, type the GO command without a numeric parameter, thus:

GO

If no mistakes were made in constructing the model that cause the simulation to abort, the computer will prompt, after some time, that the simulation is completed.
The plot produced by this run is shown in figure 64, and the output report appears in Appendix B.

Figure 64. Plot from Simple Example
7. A MORE ELABORATE EXAMPLE

In this chapter a communication system slightly more complicated than that designed in Chapter 5 and analyzed in Chapter 6 is constructed. To do this, however, requires that we introduce one further AISIM feature.

7.1 MESSAGE ROUTING SUBMODEL

When one Process is triggered by another through a CALL primitive, the called Process will execute in the same architectural node as the one that triggered it, i.e., utilize the same Resource, even if the two Processes are normally associated with different nodes. This is inconvenient in the representation of communication systems in which an activity in one hardware element causes activity in another one. AISIM therefore embodies a submodel to represent the situation in which a Process in one node triggers a Process in another node by communicating through the network architecture. This submodel consists of a collection of Processes and one Item.

The Processes that accomplish this must be placed in a project database with the commands available in the Library User Interface. The entities of this submodel need not be defined anew. For information on the use of this facility, see the AISIM User's Manual, Section 10.

7.2 DEFINING ARCHITECTURAL ELEMENTS

Consider modeling a communication system between two airbases, a headquarters and a command headquarters that communicates directly with a computer disk. Between these end-points are switches that govern the routing of messages through the system. The physical layout of this system is shown in figure 65.
For this example, the shortest paths between the nodes will be used. Therefore, subsequent to defining the architecture, method B is used to create the Legal Path Table. The resulting table is depicted in the analysis report given in Appendix C.

The operations associated with this architecture are as follows. Both airbases periodically broadcast messages to the other nodes in the system and request plans from the command headquarters.

The effect of each broadcast is to (1) stimulate processing in the HQ and CHQ and to cause the updating of information in all other nodes. Periodically an applications program in L3 requests plans from the CHQ, as do AB1 and AB2. The effect of any such request is to engage the operation of the disk that communicates directly (and only) with the CHQ.

This description of the main operations of the system implies the following more rigorous listing of the Processes that need to be defined to represent such a system. The Processes required will be:

--A Process to represent the request from the HQ to the CHQ for plans. It will execute in the HQ node and will trigger a Process in the CHQ node.
--A Process to represent the broadcast of data from AB1 and AB2 to all other nodes. This Process will execute in the nodes AB1 and AB2 and will trigger (a) an updating Processes in (a) the HQ node, (b) the CHQ node and (c) each other, i.e., a broadcast in one airbase will update information in the other.

--A Process to represent the updating activity that occurs in the CHQ, triggered by broadcasts from the airbases.

--A Process to represent the updating activity that occurs in the HQ that is triggered by broadcasts from the airbases.

--A Process to represent the updating activity in the airbases which is triggered by broadcasts from one another.

--A Process to represent the formulation of plans at the CHQ, which is triggered by requests from AB1, AB2 and HQ. This Process executes in the CHQ node and triggers another Process representing disk operation in the disk node.

--A Process to represent the operation of the disk that communicates with the CHQ node.

These descriptions can be used to generate the AISIM Process definitions found on the following pages. The Process flowcharts for each are displayed, together with annotations to clarify the rationale for the steps that might otherwise be obscure.
AIDBASE REQUEST FOR PLANS REPORT FROM CHQ

START

AB-REQ

MSG

RETURN

The Process is given the value of the variable MSG which happens to be an Item.

1. GIVEN

MSG

2. GIVEN

PLANS 5

PREGRES 200

CHQ MSG

3.

CALL

MRS

4.

RETURN

5.

WAIT 5

6.

END
The Process is given the variables LENGTH and DISK which resolve, respectively, to a numeric and a Resource.

The attribute of the (Resource) DISK called SPEED is assigned to the local variable V.SPEED.

The transfer time is set equal to the length of the message divided by V.SPEED.

The Resource DISK is allocated.

The value of the disk attribute SEEK is assigned to the local variable SEEKTIME.

Time is taken up finding data on the disk in the amount of SEEKTIME.
The value of the attribute (of the Resource "DISK") is assigned to the local variable "LATETIME".

Time is consumed in the amount of LATETIME to represent latency.

Time is consumed to represent transfer time in the constant amount carried by the global constant XFERTIME.

The disk is deallocated.
This process differs from HQ-DATA and CHQ-DATA only in the Node in which it takes place.

Consume 0.1 units of time to perform update processing.
REQUEST FOR PLANS FROM CHQ

1. MSG

GIVEN

START

2. RETURN

CHQ
NO

The Process is given the local variable MSG which denotes an Item.

MAKE MSG LENGTH = V.LENGTH

The attribute LENGTH of MSG is assigned to local variable V.LENGTH.

V.TIME = MULTIPLY .01 V.LENGTH

The processing time is calculated as .01 times the message length.

FORMAT CONSTANT V.TIME

The calculated amount of time is consumed for formatting plans.

GIVEN

500 DKI

CALL DISK.OP

The process to perform disk operations is invoked. PLANS does not continue until this process completed.

WAIT 10

The message length for the return message is changed to 500.

500

IS ASSIGNED TO MSG LENGTH

INCREASE MSG LENGTH

END
The process is given the value of a local variable "MSG" which in this case resolves to
an item representing a transient data element.

This primitive triggers a data update
in the CHQ.

This primitive triggers a data update
in the HQ.

The literal "$CNODE" is assigned to the
keyword "$CNODE" which denotes the Node in
which the Process is currently triggering.
"AB-DATA" takes place in two different
Nodes, AB1 and AB2.

This primitive determines whether the Node
in which the Process is presently executing
is AB1 or AB2. If it is in AB1 it branches
to the Entry primitive labeled "AB1" below.
AIR BASE STATUS BROADCAST TO ALL OTHER NODES

6
GIVEN
UPDATE 10
REQMORE 750
AB1
MSG

CALL
MRS

RETURN

NOWAIT 10

END

This Call primitive triggers a data update in the AB1 Node.

7
BRANCH TO THE END

ENTRY FROM COMPARE NODE

8
AB1

ENTRY FROM REQUEST TO AB1

9
GIVEN
UPDATE 10
REQMORE 750
AB2
MSG

CALL
MRS

RETURN

NOWAIT 10

END

This Call primitive triggers data updating in the AB2 Node.

This primitive branches to the Entry primitive labeled END below.

The Process branches to this Entry primitive if the Process is executing in the AB1 Node.

This Entry primitive is the one branched to subsequent to the calling of an updating Process in AB2.
The Process is given the value of a local variable "MSG" which in this case resolves to an Item representing a transient data element.

The attribute carried by the Item MSG called "LENGTH" is assigned to the variable "V.LENGTH".

The variable V.TIME which represents the time required to process the data from the airbases is calculated as (V.LENGTH) x (.015).

This Action primitive consumes time equal to the value of V.TIME calculated above.
AIRBASE REQUEST FOR PLANS REPORT FROM CHQ

The Process is given the value of a local variable "MSG" which in this case resolves to an item representing a transient data element.

This primitive triggers a request for plans from the CHQ for an airbase.
HQ REQUEST FOR STATUS DISPLAY FROM CHQ

1. GIVEN
   MSG
   HQ-REQ
   START
   MSG
   RETURN

2. GIVEN
   PLANS 4
   $REQRESP 200
   CHQ MSG
   CALL MRS
   WAIT 4
   END

3. This Process is given the value of a local variable "MSG" which in this case resolves to an item representing a transient data element.

4. This primitive triggers a request for plans from the CHQ for HQ.
7.3 DEFINING REMAINING MODEL ELEMENTS

The remaining components of the model must be defined. These include (1) all Resources accessed in the Processes (2) all Actions which appear as Primitives; (3) all Constants and global Variables used in the Processes; (4) Loads; and (5) a Scenario.

7.3.1 RESOURCE DEFINITIONS

All the Resources necessary to this model will have been defined automatically with default values while representing the physical layout represented in the Architecture Design Editor. Thus, if the nodes and links have the same names as in figure 65 above, the following list of Resources will already exist in the model database.

<table>
<thead>
<tr>
<th>No.</th>
<th>Resource Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>451</td>
<td>RESOURCE FOR NODE</td>
</tr>
<tr>
<td>452</td>
<td>RESOURCE FOR MORE</td>
</tr>
<tr>
<td>451.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>451.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>452.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>452.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>453.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>453.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>454.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>454.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>455.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>455.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>456.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>456.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>457.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>457.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>458.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>458.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>459.A</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>459.B</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td>460</td>
<td>COMMAND HEADQUARTERS</td>
</tr>
<tr>
<td>461</td>
<td>DISK FOR COMMAND HEADQUARTERS</td>
</tr>
<tr>
<td>462</td>
<td>HEADQUARTERS</td>
</tr>
<tr>
<td>463</td>
<td>RESOURCE FOR MODE</td>
</tr>
<tr>
<td>464</td>
<td>SWITCH BETWEEN APPOSES AND OTHER TWO SWITCHES (42)</td>
</tr>
<tr>
<td>465</td>
<td>SWITCH BETWEEN SWITCH 1 &amp; 2 AND 48</td>
</tr>
<tr>
<td>466</td>
<td>SWITCH BETWEEN SWITCH 1 &amp; 2 AND 49</td>
</tr>
</tbody>
</table>

Figure 66. Defined Resource Entities

Since these Resources are created with default values (an initial unit of 1, a maximum unit of 1, and no description), they must be edited to provide helpful descriptions and to give them attributes since attributes of these Resources are accessed in several places in the Message Routing Submodel. Descriptions and attributes can be edited before generating the architecture using the ADE's DEFINE command. See Section 2.2 of this manual.
7.3.2 FILLING IN THE ACTION DEFINITIONS. The Action Primitives in the Processes must have corresponding Action entity definitions outside the Process. The Actions used are these:

- FORMAT TIME USED TO FORMAT PLANS FROM CHQ
- LATENCY LATENCY PAUSE SUBSEQUENT TO SEEK
- ROUTECH PROCESSING DELAY TO ROUTE A MESSAGE
- SEEK SEEKING INFORMATION ON DISK
- UPDATE UPDATING INFO SINCE PREVIOUS BROADCAST TO OTHER NODES
- IFER TRANSFER INFORMATION BOUGHT ON DISK
- IFER.OH PROCESSING DELAY TO ROUTE A MESSAGE OVER A CHANNEL

Figure 67. Defined Action Entities

7.3.3 CONSTANTS AND GLOBAL VARIABLES

This model contains five global Variables (ABDRATE, ABRATE, HQRATE, TIME1 and VRATE) and one Constant (V.TRACE). Their defined values and descriptions (which explain their role in the model) are as shown in figure 68.

- ABDRATE INTERVAL RATE BETWEEN SIGNALS
- ABRATE INTERVAL RATE BETWEEN SIGNALS
- HQRATE INTERVAL RATE BETWEEN SIGNALS
- TIME1 AVERAGE SEEK TIME FOR DISK IN MILLI SECONDS
- VRATE SWITCH-OTHER NODE CHANNEL SPEED IN MBYTE
- V.TRACE DEFAULT IS NO TRACE ON

Figure 68. Defined Constant Entities

7.3.4 DEFINING LOADS AND SCENARIOS. In this model we wish to represent the several Process triggerings that are due to causes outside the system. First, the AB1 and AB2 will broadcast communications to the other nodes (which trigger updating Processes in them) every minute, by an interval scheduling method. In addition, AB1 and AB2 will issue requests for plans from the CHQ sixty times in one hour by an exponential scheduling method. We define a second Load to represent requests from the leaf-node, L3, also for plans from the CHQ. This Process will also be undertaken sixty times per hour, exponentially distributed. The Load definitions implied by these requirements are printed in appendix C.

The length of the entire Scenario is 360,000 milliseconds (one hour), which is divided into ten periods of six minutes each. To simulate the operation of the system with the worst case, we stipulate that both of the functional Loads are triggered simultaneously, at the beginning of the Scenario. In addition, as a monitoring device, we initiate the Trace Process at the beginning of the simulation run. The parameters for the Scenario implied by these requirements are printed page 16 of the analysis report in appendix C.
7.4 ANALYZING THE MODEL

To run the model through a simulation test, invoke the Analysis User Interface from the AISIM READY level. For this example, the simulation will not be interrupted at the ends of periods, nor will graphs be defined.

The analysis report obtained from a simulation run of this model appears in appendix C.
APPENDIX A

TERMINAL PROFILES FOR FORMS
Figure 69. Terminal Profiles for Forms

Figure 69 describes the function keys which are used to move through the forms on each terminal. Following is a description of the ways in which a user can move through a form. These movements correspond to the column headings in the figure.

**UP** - If the cursor is in a block of fields, such as Resource attributes, the cursor will move up to the field above it. If the cursor is in a single field or at the top of a block, the cursor will move to the end of the next field above it. If there are no fields above it, the cursor will wrap to the end of the last field in the form.

**DOWN** - If the cursor is in a block of fields, such as Resource attributes, the cursor will move down to the field below it. If the cursor is in a single field or at the bottom of a block, the cursor will move to the beginning of the next field below it. If there are no fields below it, the cursor will wrap to the beginning of the first field in the form.

**LEFT** - The cursor will move one position to the left in the current field. If the cursor is at the beginning of a field, it will move to the end of the previous field. If the cursor is at the top of the form, it will wrap to the end of the last field in the form.

**RIGHT** - The cursor will move one position to the right in the current field. If the cursor is at the end of a field, it will move to the beginning of the next field. If the cursor is at the end of the form, it will wrap to the beginning of the first field in the form.

**ENTER** - Cut the form and send the data in the form to be processed by the ATSIM system.

<table>
<thead>
<tr>
<th></th>
<th>UP</th>
<th>DOWN</th>
<th>LEFT</th>
<th>RIGHT</th>
<th>ENTER</th>
<th>+FIELD</th>
<th>-FIELD</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP2647A</td>
<td>F1</td>
<td>F2</td>
<td>F3</td>
<td>F4</td>
<td>F5</td>
<td>&lt;cr&gt;</td>
<td>F6</td>
</tr>
<tr>
<td>HP2648A</td>
<td>F1</td>
<td>F2</td>
<td>F3</td>
<td>F4</td>
<td>F5</td>
<td>&lt;cr&gt;</td>
<td>F6</td>
</tr>
<tr>
<td>HP2623</td>
<td>F1</td>
<td>F2</td>
<td>F3</td>
<td>F4</td>
<td>F5</td>
<td>&lt;cr&gt;</td>
<td>F6</td>
</tr>
<tr>
<td>TEK4105</td>
<td>F1</td>
<td>F2</td>
<td>F3</td>
<td>F4</td>
<td>F5</td>
<td>&lt;cr&gt;</td>
<td>F6</td>
</tr>
<tr>
<td>VT100</td>
<td>↑</td>
<td>↓</td>
<td>&lt;--</td>
<td>--&gt;</td>
<td>PF1</td>
<td>&lt;cr&gt;</td>
<td>PF2</td>
</tr>
</tbody>
</table>
+FIELD - Move the cursor to the beginning of the next field in the form. If the cursor is at the end of the form, it will wrap to the top of the form.

-FIELD - Move the cursor to the end of the previous field in the form. If the cursor is at the top of the form, it will wrap to the end of the last field in the form.
APPENDIX B

SIMULATION REPORT FOR WORKING EXAMPLE
GLOBAL CONSTANT DEFINITION.....

CONSTANT INITIAL
MNEMONIC VALUE COMMENT
============================================================================

TABLE DEFINITION.....

GLOBAL VARIABLE DEFINITION.....

VARIABLE INITIAL
MNEMONIC VALUE COMMENT
============================================================================
GAMMA1 700
GAMMA2 .002

ITEM DEFINITION.....

ITEM DESCRIPTION
--------------------
MSG MSG TO BE TRANSMITTED
ATTR. INITIAL NAME VALUE
--------------------
LENGTH $LENGTH

QUEUE DEFINITION.....

QUEUE MAXIMUM
MNEMONIC SIZE COMMENT
--------------------
BUFFER INFINITE BUFFER ON WHICH MESSAGES ARE STORED

RESOURCE DEFINITION.....

RESOURCE TOTAL INITIAL
MNEMONIC # UNITS # UNITS DESCRIPTION
--------------------
BUFF1 1 1 RESOURCE ASSOCIATED WITH BUFFER
ARCHITECTURE LEGAL PATH DEFINITION

FROM DEVICE TO DEVICE NEXT VIA DEVICE LINK

ACTION DEFINITION

ACTION

MNEMONIC CLASS COMMENT

READ-MSG MACHINE READ A MESSAGE
SENDING MACHINE TRANSMIT A MESSAGE

PROCESS DEFINITION

ENTRY OPCODE PARM PARM PARM COMMENT

START NO
TEST BUF1 ABORT TEST FOR BUFFER USE
REMOVE FIRST MSG BUFFER REMOVE BY FIFO DISCIPLINE
COMPARE MSG EQ ABORT WHEN MSG=0 BUFFER IS EMPTY
ASSIGN MSG LENGTH MSG LENGTH IS READ
EVAL MU MULTIPLY CALCULATE RECEPTION TIME
READ-MSG ALPHA GAMMA2 TIME TO PROCESS MESSAGE
DESTROY MU MSG ELIMINATED FROM SYSTEM
ABORT ENTRY ENTRY ENTER FROM COMPARE & TEST

LOCAL VARIABLES OF PROCESS RECEIVE

1 BUF1 (R) 2 MSG (I) 3 BUFFER (Q) 4 ALPHA
5 MU 6 READ-MSG (A)

PROCESS

MNEMONIC DESCRIPTION
TRANSMIT TRANSMITTING MESSAGES TO RECEIVER

ENTRY  OPCODE  PARW  PARW  PARW  COMMENT
-------- -------- -------- --------  ---------------------------
START    NO      NO      NO      ALLOCATE BUF1
ALLOC    BUF1    1      ALL      ALLOCATE BUF1
$PRIOR    MSG      RANDOM     INTRODUCE MSG INTO SYSTEM
CREATE    ALPHA    RANDOM     GENERATE RANDOM NUMBER
EVAL     ALPHA    MULTIPLY    TWICE AVERAGE TIMES ALPHA
ASSIGN    MSG      LENGTH     SET MESSAGE LENGTH
EVAL     WMU      MULTIPLY    CALCULATE TRANSMIT TIME
SENDING  MSG      LAST       BUFFER  STORE MSG ON BUFFER
FILE MSG LAST BUFFER  STORE MSG ON BUFFER
DEALLOC  BUF1    1      ALL      RELEASE RESOURCE BUF1
END

LOCAL VARIABLES OF PROCESS TRANSMIT
------------------------------------------------------------------------------------
1 BUF1 (R) 2 MSG (I) 3 ALPHA (A) 4 MU (Q)
5 SENDING (A) 6 BUFFER (Q)

LOAD DEFINITION.....

MNEUMONIC DESCRIPTION
---------- -------------------------------

L1 LOAD NODES
---------- -------------------------------

PROCESS SCHEDULE
MNEUMONIC MAX # METHOD MEAN DELTA PRIORITY
---------- ---------- ---------- ---------- ---------- ----------
TRANSMIT 300  POISSON 0

LOAD MNEUMONIC DESCRIPTION
---------- -------------------------------

L1 LOAD NODES
---------- -------------------------------


LOAD NODES

PROCESS SCHEDULE
MNEMONIC  WAX  METHOD  MEAN  DELTA  PRIORITY

RECEIVE  200  INTERVAL  0.00001

SCENARIO DEFINITION....

SCENARIO
MNEMONIC  DESCRIPTION

PERIOD LENGTH

100

PERIOD  PERIOD  PERIOD  PERIOD  PERIOD  PERIOD  PERIOD
MNEMONIC  MNEMONIC  MNEMONIC  MNEMONIC  MNEMONIC  MNEMONIC

1  2  3  4  5  6

TRIGGER TIME TO SCHEDULE TRIGGER TIME TO SCHEDULE
MNEMONIC  SCHEDULE PRIORITY MNEMONIC  SCHEDULE PRIORITY

L1  100  0  L11  100  0
L2  300  0  L22  300  0
L3  500  0  L33  500  0

0 ERRORS WERE DETECTED DURING MODEL INITIALIZATION
SIMULATION TIME = 600.00000 UNITS

VARIABLE REPORT

NUMERIC VARIABLES...

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>TOTAL SAMPLES</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAMMA1</td>
<td>1</td>
<td>700.000</td>
<td>700.000</td>
<td>0.000</td>
<td>700.000</td>
<td>700.000</td>
</tr>
<tr>
<td>GAMMA2</td>
<td>3</td>
<td>0.002</td>
<td>0.002</td>
<td>0.000</td>
<td>0.001</td>
<td>0.002</td>
</tr>
</tbody>
</table>

NON-NUMERIC VARIABLES...

<table>
<thead>
<tr>
<th>CURRENT VARIABLE</th>
<th>CURRENT TYPE</th>
<th>CURRENT VALUE</th>
</tr>
</thead>
</table>

| GAMMA1 | | |
| GAMMA2 | | |
SIMULATION TIME = 600.00000 UNITS

ITEM REPORT

<table>
<thead>
<tr>
<th>ITEM NAME</th>
<th>NUMBER CREATED</th>
<th>NUMBER DESTROYED</th>
<th>TIME IN SYSTEM</th>
<th>TIME IN SYSTEM</th>
<th>TIME IN SYSTEM</th>
<th>TIME IN SYSTEM</th>
<th>TIME IN SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>500</td>
<td>000</td>
<td>30.50</td>
<td>200.80</td>
<td>142.90</td>
<td>33.70</td>
<td></td>
</tr>
</tbody>
</table>
SIMULATION TIME = 600.00000 UNITS

QUEUE REPORT

<table>
<thead>
<tr>
<th>QUEUE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFER</td>
<td>FILED ON 500</td>
<td>96.000</td>
<td>126.480</td>
<td>107.531</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>REMOVED FROM 500</td>
<td>141.755</td>
<td>33.713</td>
<td>67.932</td>
<td>199.654</td>
</tr>
</tbody>
</table>

| TASKS BLOCKED | 0 |
| TASKS RESUMED | 0 |
| # BEING BLOCKED | 0 |
| TIME BLOCKED | 0 |
SIMULATION TIME = 600.0000 UNITS

ACTION REPORT

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% TIME OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ-MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL TIME</td>
<td>500</td>
<td>0.875</td>
<td>0.410</td>
<td>0.001</td>
<td>1.398</td>
<td>56.244</td>
</tr>
<tr>
<td>DELAY TIME</td>
<td>500</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>SENDING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL TIME</td>
<td>590</td>
<td>0.490</td>
<td>0.340</td>
<td>0.000</td>
<td>1.398</td>
<td>48.722</td>
</tr>
<tr>
<td>DELAY TIME</td>
<td>590</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>
SIMULATION TIME = 000.00000 UNITS

PROCESS REPORT

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>TOTAL SAMPLES</th>
<th>SUM</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECEIVE</td>
<td>TOTAL</td>
<td>1200</td>
<td>337.465</td>
<td>0.281</td>
<td>0.425</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>PROCESS WAIT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>RESOURCE WAIT</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

TOTAL # AUTO # CALL # OF # NOT # TIMES SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND

<table>
<thead>
<tr>
<th>ITEM</th>
<th>CREATED</th>
<th>RECEIVED</th>
<th>SENT</th>
<th>DESTR'D</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

PROCESS HOLDING TIME

<table>
<thead>
<tr>
<th>ITEM</th>
<th># SMPLS</th>
<th>SUM</th>
<th>MEAN</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>STD DEV</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>500</td>
<td>0.87</td>
<td>0.00</td>
<td>1.40</td>
<td>0.41</td>
<td></td>
</tr>
</tbody>
</table>

PROCESS DESCRIPTION

<table>
<thead>
<tr>
<th>COUNT ENTRY</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1200</td>
<td>START</td>
<td>NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td>TEST</td>
<td>BUF1</td>
<td>ABORT</td>
<td></td>
<td></td>
<td>TEST FOR BUFFER USE</td>
</tr>
<tr>
<td>1200</td>
<td>REMOVE</td>
<td>FIRST</td>
<td>MSG</td>
<td>BUFFER</td>
<td></td>
<td>REMOVE BY FIFO DISCIPLINE</td>
</tr>
<tr>
<td>1200</td>
<td>COMPARE</td>
<td>MSG</td>
<td>EQ</td>
<td></td>
<td></td>
<td>WHEN MSG=0 BUFFER IS EMPTY</td>
</tr>
<tr>
<td>1200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>ASSIGN</td>
<td>MSG</td>
<td></td>
<td>LENGTH</td>
<td></td>
<td>MSG LENGTH IS READ</td>
</tr>
<tr>
<td>500</td>
<td>EVAL</td>
<td>MU</td>
<td></td>
<td>MULTIPLY</td>
<td></td>
<td>CALCULATE RECEPTION TIME</td>
</tr>
<tr>
<td>500</td>
<td></td>
<td>ALPHA</td>
<td></td>
<td>GAMMA2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>READ-MSG</td>
<td>CONSTANT</td>
<td>MU</td>
<td></td>
<td></td>
<td>TIME TO PROCESS MESSAGE</td>
</tr>
<tr>
<td>500</td>
<td></td>
<td>DESTROY</td>
<td>MSG</td>
<td></td>
<td></td>
<td>MSG ELIMINATED FROM SYSTEM</td>
</tr>
<tr>
<td>1200</td>
<td>ABORT</td>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td>ENTER FROM COMPARE &amp; TEST</td>
</tr>
<tr>
<td>1200</td>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOTAL PROCESS

<table>
<thead>
<tr>
<th>TOTAL SAMPLES</th>
<th>SUM</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
</table>

Page 97
<table>
<thead>
<tr>
<th>INDEX</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>596</td>
<td>START</td>
<td>NO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>596</td>
<td>ALLOC</td>
<td>BUF1</td>
<td>1</td>
<td>ALL</td>
<td>ALLOCATE BUF1</td>
</tr>
<tr>
<td>596</td>
<td>CREATE</td>
<td>MSG</td>
<td></td>
<td></td>
<td>INTRODUCE MSG INTO SYSTEM</td>
</tr>
<tr>
<td>596</td>
<td>EVAL</td>
<td>ALPHA</td>
<td>RANDOM</td>
<td></td>
<td>GENERATE RANDOM NUMBER</td>
</tr>
<tr>
<td>596</td>
<td>EVAL</td>
<td>ALPHA</td>
<td>MULTIPLY</td>
<td></td>
<td>TWICE AVERAGE TIMES ALPHA</td>
</tr>
<tr>
<td>596</td>
<td>ASSIGN</td>
<td>ALPHA</td>
<td>LENGTH</td>
<td></td>
<td>SET MESSAGE LENGTH</td>
</tr>
<tr>
<td>596</td>
<td>EVAL</td>
<td>MSG</td>
<td>MULTIPLY</td>
<td></td>
<td>CALCULATE TRANSMIT TIME</td>
</tr>
<tr>
<td>596</td>
<td>SENDING</td>
<td>MJ</td>
<td></td>
<td></td>
<td>TIME CONSUMED TRANSMITTING</td>
</tr>
<tr>
<td>596</td>
<td>FILE</td>
<td>MSG</td>
<td>LAST</td>
<td>BUFFER</td>
<td>STORE MSG ON BUFFER</td>
</tr>
<tr>
<td>596</td>
<td>DEALLOC</td>
<td>BUF1</td>
<td>1</td>
<td></td>
<td>RELEASE RESOURCE BUF1</td>
</tr>
</tbody>
</table>
APPENDIX C

SIMULATION REPORT FOR ELABORATE EXAMPLE
GLOBAL CONSTANT DEFINITION....

CONSTANT INITIAL
MNEMONIC VALUE COMMENT

TABLE DEFINITION....

GLOBAL VARIABLE DEFINITION....

VARIABLE INITIAL
MNEMONIC VALUE COMMENT

ITEM DEFINITION....

ITEM    DESCRIPTION
---------    ---------------------------------
MSG        MESSAGE FOR INTERNODE COMMUNICATION

ATTR.    INITIAL VALUE
---------    ---------------------------------
NAME       MSG

QUEUE DEFINITION....

QUEUE MAXIMUM
<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL</th>
<th>INITIAL</th>
<th># UNITS</th>
<th># UNITS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>NODE</td>
<td>RESOURCE FOR NODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AB2</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>NODE</td>
<td>RESOURCE FOR NODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH1.A</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>CHANNEL CONNECTOR</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH1.B</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>CHANNEL CONNECTOR</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH2.A</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>CHANNEL CONNECTOR</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH2.B</td>
<td>1</td>
<td>1</td>
<td>REST</td>
<td>CHANNEL CONNECTOR</td>
<td>RESOURCE FOR CHANNEL CONNECTOR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CH3.A  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH3.B  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH4.A  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH4.B  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH5.A  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH5.B  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  0.4069

CH6.A  1
ATTR. INITIAL
NAME VALUE
====== ======
COST  0
RATE  VRATE
<table>
<thead>
<tr>
<th>PAGE 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH8.B 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>CH7.A 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>CH7.B 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>CH8.B 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>CH9.A 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>CH9.B 1</td>
</tr>
<tr>
<td>ATTR.</td>
</tr>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>COST</td>
</tr>
<tr>
<td>Page 104</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td><strong>CMQ</strong></td>
</tr>
</tbody>
</table>
| 1 | 1 | COMMAND HEAD-QUARTERS  
| ATTR. | INITIAL NAME | VALUE | costs | M.ROUTE | 8  
| LAT**DELTA** | 16 | LATENCY | 16 | M.ROUTE | 0 | SEEK | TIME1 | SPEED | 20000 |  
| **HQ** |  
| 1 | 1 | HEAD-QUARTERS  
| ATTR. | INITIAL NAME | VALUE | costs | M.ROUTE | 8  
| **L3** |  
| 1 | 1 | RESOURCE FOR NODE  
| ATTR. | INITIAL NAME | VALUE | costs | M.ROUTE | 8  
| **SW1** |  
| 1 | 1 | SWITCH BETWEEN AIRBases AND OTHER TWO SWITCHES (1&2)  
| ATTR. | INITIAL NAME | VALUE | costs | M.ROUTE | 8  
| **SW2** |  
| 1 | 1 | SWITCH BETWEEN SWITCH 1 & 3 AND HQ  
| ATTR. | INITIAL NAME | VALUE | costs | M.ROUTE | 8  
| **SW3** |  
| 1 | 1 | SWITCH BETWEEN SWITCH 1 & 2 AND CHQ  
<p>| ATTR. | INITIAL |</p>
<table>
<thead>
<tr>
<th>FROM DEVICE</th>
<th>TO DEVICE</th>
<th>NEXT DEVICE</th>
<th>VIA LINK</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>AB2</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>CHQ</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>DK1</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>HQ</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>L3</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>SW1</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>SW2</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>SW3</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB1</td>
<td>AB1</td>
<td>SW1</td>
<td>CH1.A</td>
</tr>
<tr>
<td>AB2</td>
<td>CHQ</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>DK1</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>HQ</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>L3</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>SW1</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>SW2</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>AB2</td>
<td>SW3</td>
<td>SW1</td>
<td>CH2.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>AB1</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>AB2</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>DK1</td>
<td>DK1</td>
<td>CH9.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>HQ</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>L3</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>SW1</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>SW2</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>CHQ</td>
<td>SW3</td>
<td>SW3</td>
<td>CH8.A</td>
</tr>
<tr>
<td>DK1</td>
<td>AB1</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>AB2</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>CHQ</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>HQ</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>L3</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>SW1</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>SW2</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>DK1</td>
<td>SW3</td>
<td>CHQ</td>
<td>CH9.B</td>
</tr>
<tr>
<td>HQ</td>
<td>AB1</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
<tr>
<td>HQ</td>
<td>AB2</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
<tr>
<td>HQ</td>
<td>CHQ</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
<tr>
<td>HQ</td>
<td>DK1</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
<tr>
<td>HQ</td>
<td>L3</td>
<td>L3</td>
<td>CH6.A</td>
</tr>
<tr>
<td>HQ</td>
<td>SW1</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
<tr>
<td>HQ</td>
<td>SW2</td>
<td>SW2</td>
<td>CH7.A</td>
</tr>
</tbody>
</table>
ACTION DEFINITION.....

ACTION ACTION

WNEMONIC CLASS COMMENT

FORMAT MACHINE TIME USED TO FORMAT PLANS FROM CHQ
LATENCY MACHINE LATENCY PAUSE SUBSEQUENT TO SEEK
ROUTE.OH CPU PROCESSING DELAY TO ROUTE A MESSAGE
SEEK MACHINE SEEKING INFORMATION ON DISK
UPDATE AIRBASE UPDATING INFO SINCE PREVIOUS BROADCAST TO OTHER NODES
XFER MACHINE TRANSFER INFORMATION SOUGHT ON DISK
XFER.OH CHANNEL PROCESSING DELAY TO ROUTE A MESSAGE OVER A CHANNEL

PROCESS DEFINITION.....
### AB-DATA

**DESCRIPTION**

AIR BASE STATUS BROADCAST TO ALL OTHER NODES

<table>
<thead>
<tr>
<th>ENTRY</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETURN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>WRS</td>
<td>NOWAIT</td>
<td>10</td>
<td>$REQORE</td>
<td>PROCESS REQUEST TO CHQ</td>
</tr>
<tr>
<td>GIVEN</td>
<td>CHQ-DATA</td>
<td>10</td>
<td>CHQ</td>
<td>MSG</td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>WRS</td>
<td>NOWAIT</td>
<td>10</td>
<td>$REQORE</td>
<td>PROCESS REQUEST TO HQ</td>
</tr>
<tr>
<td>GIVEN</td>
<td>HQ-DATA</td>
<td>10</td>
<td>HQ</td>
<td>MSG</td>
<td></td>
</tr>
<tr>
<td>ASSIGN</td>
<td>$CNODE</td>
<td></td>
<td></td>
<td></td>
<td>CURRENT NODE</td>
</tr>
<tr>
<td>COMPARE</td>
<td>CNODE</td>
<td>EQ</td>
<td></td>
<td></td>
<td>TEST FOR CURRENT NODE</td>
</tr>
<tr>
<td>CALL</td>
<td>AB1</td>
<td>NOWAIT</td>
<td>10</td>
<td>$REQORE</td>
<td>PROCESS REQUEST TO AB1</td>
</tr>
<tr>
<td>GIVEN</td>
<td>ABUPDTE</td>
<td>10</td>
<td>AB1</td>
<td>MSG</td>
<td></td>
</tr>
<tr>
<td>BRANCH</td>
<td>END</td>
<td>100</td>
<td></td>
<td></td>
<td>BRANCH TO THE END</td>
</tr>
<tr>
<td>AB1</td>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td>ENTRY FROM COMPARE NODE</td>
</tr>
<tr>
<td>CALL</td>
<td>WRS</td>
<td>NOWAIT</td>
<td>10</td>
<td>$REQORE</td>
<td>PROCESS REQUEST TO AB2</td>
</tr>
<tr>
<td>GIVEN</td>
<td>ABUPDTE</td>
<td>10</td>
<td>AB2</td>
<td>MSG</td>
<td></td>
</tr>
<tr>
<td>END</td>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td>ENTRY FROM REQUEST TO AB1</td>
</tr>
</tbody>
</table>

**LOCAL VARIABLES OF PROCESS AB-DATA**

| 1 MSG  | (I) | 2 WRS  | (P) | 3 CHQ-DATA | (P) | 4 CHQ  | (R) | 5 HQ-DATA | (P) | 6 HQ    | (R) | 7 CNODE | (R) | 8 AB1   | (R) |

### AB-REQ

**DESCRIPTION**

AISBASE REQUEST FOR PLANS REPORT FROM CHQ

<table>
<thead>
<tr>
<th>ENTRY</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETURN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>WRS</td>
<td>WAIT</td>
<td>6</td>
<td></td>
<td>PROCESS REQUEST TO CHQ</td>
</tr>
</tbody>
</table>
LOCAL VARIABLES OF PROCESS AB-REQ

1 MSG (I) 2 WRS (P) 3 PLANS (P) 4 CHQ (R)

ENTRY        OPCODE  PARM  PARM  PARM  COMMENT

START        ALL     NO
GIVEN        MSG     CNODE   SET INTERNAL NODE CURRENT
ASSIGN       MSG     TNODE   GET DESTINATION NODE
ASSIGN       $NXTNODE TO.NODE   SET NEXT NODE TO DESTN
ASSIGN       $CHANNEL TO.NODE   GET CHANNEL TO NEXT NODE
ALLOC       CHANNEL 1   ALL   OBTAIN CHANNEL FOR XFER
ASSIGN       CHANNEL   RATE   WHAT IS CHANNEL RATE?
ASSIGN       MSG       VLENGTH MESSAGE LENGTH

LOCAL VARIABLES OF PROCESS ABUPDATE

1 MSG (I) 2 UPDATE (A)

ENTRY        OPCODE  PARM  PARM  PARM  COMMENT

START        ALL     NO
GIVEN        MSG     CNODE   SET INTERNAL NODE CURRENT
ASSIGN       MSG     TNODE   GET DESTINATION NODE
ASSIGN       $NXTNODE TO.NODE   SET NEXT NODE TO DESTN
ASSIGN       $CHANNEL TO.NODE   GET CHANNEL TO NEXT NODE
ALLOC       CHANNEL 1   ALL   OBTAIN CHANNEL FOR XFER
ASSIGN       CHANNEL   RATE   WHAT IS CHANNEL RATE?
ASSIGN       MSG       VLENGTH MESSAGE LENGTH
EVAL VM.OVHD MULTIPLY VSPEED VLENGTH CALCULATE TRANSFER TIME
XFER.OH CONSTANT VM.OVHD DELAY DUE TO TRANSFER TIME
ASSIGN NXT.NODE MSG CNODE MSG RESIDES IN NEXT NODE
ASSIGN NXT.NODE $CNODE SET INTERNAL NODE REGISTER
DEALLOC CHANNEL 1 FREE UP CHANNEL AFTER XFER
CALL NODEPROC WAIT 0 ROUTE MESSAGE TO NEXT NODE
GIVEN MSG END

LOCAL VARIABLES OF PROCESS CHANPROC
==========================================================================
1 MSG (I) 2 TO.NODE 3 NXT.NODE 4 CHANNEL 5 VSPEED 6 VLENGTH 7 VM.OVHD 8 XFER.OH (A)
9 NODEPROC (P)

PROCESS
MNEMONIC DESCRIPTION
==========================================================================
CHQ-DATA CHQ GETS MESSAGE, FORMULATES RESPONSE, AND REPLIES

ENTRY OPCODE PARM PARM PARM COMMENT
==========================================================================
START NO
GIVEN MSG RETURN MSG
ASSIGN MSG LENGTH MAKE MSG-LENGTH = V.LENGTH
EVAL V.LENGTH MULTIPLY EVALUATE MSG PROCESS TIME
UPDATE .015 V.LENGTH .015 V.LENGTH PROCESSING TIME CONSUMED
END

LOCAL VARIABLES OF PROCESS CHQ-DATA
==========================================================================
1 MSG (I) 2 V.LENGTH 3 V.TIME 4 UPDATE (A)

PROCESS
MNEMONIC DESCRIPTION
==========================================================================
DESTPROC PROCESSING AT DESTINATION OF MESSAGE

ENTRY OPCODE PARM PARM PARM COMMENT
==========================================================================
START ALL NO
GIVEN MSG
ASSIGN MSG C.NODE CURRENT NODE
COMPARE MSG TYPE EQ IF RESPONSE, DESTROY
ALLOC C.NODE 1 ALL ALLOCATE CURRENT NODE
ASSIGN MSG RPROC PROCESS EXECUTE THE CALLED PROCESS
ASSIGN MSG RPROC PRI SET PRIORITY FOR REQ FUNC
CALL PROCESS WAIT PRIORITY WAIT UNTIL COMPLETE
GIVEN MSG
RETURN MSG
DEALLOC C.NODE 1 DEALLOCATE CURRENT NODE
COMPARE MSG TYPE EQ NO RESPONSE REQ -> DESTROY
ASSIGN $RESP MSG TYPE CHANGE MSG RESPONSE TYPE
ASSIGN MSG FNODE SWITCH FROM AND TO NODEs
ASSIGN MSG TNODE CURRENT NODE IS FROM NODE
ASSIGN MSG FNODE
CALL CHANPROC WAIT 0 RETURN MESSAGE TO ORIGIN
GIVEN MSG BRANCH END 100
DESTROY ENTRY
DESTROY MSG TERMINATE MESSAGE AT DEST
END ENTRY
END

LOCAL VARIABLES OF PROCESS DESTPROC

<table>
<thead>
<tr>
<th>MSG (I)</th>
<th>2 C.NODE</th>
<th>3 PROCESS</th>
<th>4 PRIORITY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PROCESS MNEMONIC DESCRIPTION

<table>
<thead>
<tr>
<th>DISK.OP</th>
<th>OPERATION OF DISK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ENTRY OPCODE PARM PARM PARM PARM COMMENT

<table>
<thead>
<tr>
<th>START</th>
<th>NO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GIVEN</td>
<td>LENGTH DISK</td>
<td>MAKE DISK SPEED = V.SPEED</td>
</tr>
<tr>
<td>ASSIGN</td>
<td>DISK</td>
<td>SPEED</td>
</tr>
<tr>
<td>EVAL</td>
<td>XFERTIME DIVIDE</td>
<td>TRANSFER TIME CALCULATED</td>
</tr>
<tr>
<td>ALLOC</td>
<td>DISK</td>
<td>1</td>
</tr>
</tbody>
</table>
PAGE 12

ASSIGN DISK SEEK MAKE SEEKTIME = SEEK
SEEK UNIFORM SEEKTIME SEEKTIME TIME FOR SEEK IS CONSUMED
ASSIGN DISK LATENCY MAKE DISK LATENCY = LATETIME
LATETIME UNIFORM LATETIME LATETIME TIME CONSUMED FOR LATENCY
XFER CONSTANT XFERTIME TRANSFER TIME CONSUMED
DEALLOC DISK 1 DISK RESOURCE DEALLOCATED

LOCAL VARIABLES OF PROCESS DISK.OP
=================================================================================================================================
1 LENGTH 2 DISK 3 V.SPEED 4 XFERTIME
5 SEEKTIME 6 SEEK (A) 7 LATETIME 8 LATENCY (A)
9 XFER (A)

PROCESS MNEMONIC DESCRIPTION
=================================================================================================================================
HQ-DATA HQ GETS MESSAGE, FORMULATES RESPONSE, AND REPLIES

ENTRY OPCODE PARM PARM PARM COMMENT
=================================================================================================================================
START NO
GIVEN MSG
RETURN MSG
ASSIGN MSG LENGTH MAKE MSG-LENGTH = V.LENGTH
EVAL V.TIME MULTIPLY EVALUATE MSG PROCESS TIME .015 V.LENGTH
UPDATE CONSTANT V.TIME PROCESSING TIME CONSUMED
END

LOCAL VARIABLES OF PROCESS HQ-DATA
=================================================================================================================================
1 MSG (I) 2 V.LENGTH 3 V.TIME 4 UPDATE (A)

PROCESS MNEMONIC DESCRIPTION
=================================================================================================================================
HQ-REQ HQ REQUEST FOR STATUS DISPLAY FROM CHQ

ENTRY OPCODE PARM PARM PARM COMMENT
=================================================================================================================================
START NO
GIVEN MSG
RETURN MSG
CALL MRS WAIT 4 MAKES I/O REQUEST TO CHQ
GIVEN PLANS 4 $REQRESP
END

LOCAL VARIABLES OF PROCESS HQ-REQ
==============================================
1 MSG (I) 2 MRS (P) 3 PLANS (P) 4 CHQ (R)

PROCEDURE
MNEMONIC DESCRIPTION
============= ================
MRS GENERATE A PROCESS REQUEST MESSAGE AND INITIATE I/O

ENTRY OPCODE PARM PARM PARM COMMENT
======== ====== ====== ====== ================
START ALL NO NO NO EXECUTE MESSAGE
GIVEN PROCESS PRIORITY MSG.TYPE
CREATE MSG MSG.MSGLNTH TO.NODE MSG
ASSIGN MSG.MSGLNTH LENGTH
ASSIGN MSG.MSG PRIORITY
ASSIGN TO.NODE MSG RPROC
ASSIGN MSG.MSGLNTH TYPE
ASSIGN MSG.MSGLNTH NPROC
CALL NPROC WAIT 0 EXECUTE SERVICING OF MSG
END

LOCAL VARIABLES OF PROCESS MRS
==============================================
1 PROCESS (X) 2 PRIORITY (X) 3 MSG.TYPE 4 MSG.MSGLNTH
6 TO.NODE 5 MSG (I) 7 NPROC (P)

PROCEDURE
MNEMONIC DESCRIPTION
============= ================
NPROC NODAL PROCESSING AND ROUTING

ENTRY OPCODE PARM PARM PARM PARM COMMENT
======== ====== ====== ====== ================
START ALL NO NO NO
GIVEN MSG
ASSIGN MSG CNODE INDICATE CURRENT NODE
ASSIGN C.NODE M.ROUTE PROCESSING RATE OF NODE
ASSIGN MSG RT.OVHD RT. O V H D
ASSIGN MSG MSG.LNTH GET MESSAGE LENGTH
EVAL MSG.LNTH MSG.LNTH MULTIPY COMPUTE PROCESSING DELAY
ALLOC C.NODE 1 ALL ALLOCATE CURRENT NODE
ROUTE.OH 1 PRIORITY DELAY FOR ROUTING
DEALLOC C.NODE 1 RELEASE C.NODE TO OTHERS
COMPARE C.NODE MSG EQ IS MSG AT DESTINATION?
MSG CNODE CHANPROC WAIT 0 FORWARD MSG TO CHANNEL
CALL MSG END 100
CONTROL ENTRY CALL DESTPROC WAIT 0 MESSAGE AT DESTINATION
GIVEN MSG CONTEXT SWITCH MESSAGE
END ENTRY

LOCAL VARIABLES OF PROCESS NODEPROC
========================================================================
  1 MSG (I)  2 C.NODE  3 RT.OVHD  4 MSG.LNTH
  5 OVERHEAD  6 ROUTE.OH (A)  7 CHANPROC (P)  8 DESTPROC (P)

PROCESS

MNEMONIC DESCRIPTION
====== ====== ==================================================================
PLANS REQUEST FOR PLANS FROM CHQ

ENTRY OPCODE PARM PARM PARM COMMENT
===== ====== ====== ====== ====== ====== ======
START CHQ NO
GIVEN MSG NO
RETURN MSG
ASSIGN MSG V.LENGTH MAKE MSG LENGTH = V.LENGTH
EVAL V.TIME V.LENGTH EVALUATE MSG PROCESS TIME
FORMAT V.TIME V.LENGTH TIME USED TO FORMAT PLANS
CALL DISK.OP 10 CALLING PROCESS DISK.OP
GIVEN 500 DKI
ASSIGN 500 MSG INCREASE MSG LENGTH
END MSG LENGTH
LOCAL VARIABLES OF PROCESS PLANS

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACE</td>
<td>TURN ON TRACE OUTPUT</td>
<td></td>
</tr>
</tbody>
</table>

ENTRY CODE PARM PARM PARM COMMENT

- START ALL NO
- COMPARE Y-TRACE EQ TEST IF FLAG SET FOR TRACE
- TRACE ON
- NOTRACE ENTRY END

LOAD DEFINITION.....

LOAD

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABLOAD</td>
<td>COMMUNICATIONS FROM AIRBASES</td>
</tr>
</tbody>
</table>

LOAD NODES

- AB1 AB2

PROCESS SCHEDULE

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Max #</th>
<th>Method</th>
<th>Mean</th>
<th>Delta</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB-LOAD</td>
<td>60</td>
<td>INTERVAL</td>
<td>AB-LOAD RATE</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>AB-REQ</td>
<td>60</td>
<td>EXPONENT</td>
<td>AB-LOAD RATE</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

LOAD

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQLOAD</td>
<td>REQUEST DATA FROM CHQ</td>
</tr>
</tbody>
</table>

LOAD NODES

- L3

PROCESS SCHEDULE

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Max #</th>
<th>Method</th>
<th>Mean</th>
<th>Delta</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>MQ-LOAD</td>
<td>60</td>
<td>EXPONENT</td>
<td>MQ-LOAD RATE</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
### Scenario Definition

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEST01</td>
<td>SCENARIO FOR MINI MIKTR 1</td>
</tr>
</tbody>
</table>

**Period Length**: 360000

<table>
<thead>
<tr>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
<td>MNEMONIC</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Trigger Time to Schedule Priority**

<table>
<thead>
<tr>
<th>ALOAD</th>
<th>HLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Trace**

<table>
<thead>
<tr>
<th>Trace</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

**** 0 errors were detected during model initialization
<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>TOTAL</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STANDARD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB crate</td>
<td>1</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>MOR rate</td>
<td>1</td>
<td>72000.0</td>
<td>72000.0</td>
<td>72000.0</td>
<td>72000.0</td>
<td>72000.0</td>
</tr>
<tr>
<td>T$i$me</td>
<td>1</td>
<td>38.0000</td>
<td>38.0000</td>
<td>38.0000</td>
<td>38.0000</td>
<td>38.0000</td>
</tr>
<tr>
<td>VR rate</td>
<td>1</td>
<td>1.628</td>
<td>1.628</td>
<td>1.628</td>
<td>1.628</td>
<td>1.628</td>
</tr>
</tbody>
</table>
PAGE 20

SIMULATION TIME = 3600000.0000 UNITS

RESOURCE REPORT

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.513</td>
<td>0.500</td>
<td>0.00</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>4694.304</td>
<td>6663.327</td>
<td>0.00</td>
<td>22000.00</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># BUSY</td>
<td>0.00</td>
<td>0.487</td>
<td>0.500</td>
<td>0.00</td>
<td>1.000</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>4221.690</td>
<td>2309.318</td>
<td>0.00</td>
<td>6000.00</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># WAITING</td>
<td>0.00</td>
<td>0.541</td>
<td>0.973</td>
<td>0.00</td>
<td>5.000</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td>4694.304</td>
<td>6663.327</td>
<td>0.00</td>
<td>22000.00</td>
<td></td>
</tr>
</tbody>
</table>

CURRENTLY ALLOCATED TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.513</td>
<td>0.500</td>
<td>0.00</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>4329.800</td>
<td>5316.789</td>
<td>0.00</td>
<td>22000.00</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># BUSY</td>
<td>0.00</td>
<td>0.487</td>
<td>0.500</td>
<td>0.00</td>
<td>1.000</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>4221.690</td>
<td>2309.317</td>
<td>0.00</td>
<td>6000.00</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>415</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># WAITING</td>
<td>0.00</td>
<td>0.499</td>
<td>0.911</td>
<td>0.00</td>
<td>5.000</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td>4329.800</td>
<td>5316.789</td>
<td>0.00</td>
<td>22000.00</td>
<td></td>
</tr>
</tbody>
</table>
CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY
WAITING: NONE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT...</th>
<th>MEAN......</th>
<th>STD DEV...</th>
<th>MINIMUM...</th>
<th>MAXIMUM...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH1.A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.935</td>
<td>0.247</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>237</td>
<td>0.</td>
<td>0.086</td>
<td>0.247</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>237</td>
<td>0.</td>
<td>994.094</td>
<td>309.263</td>
<td>325.500</td>
<td>1220.750</td>
</tr>
<tr>
<td># BUSY</td>
<td>237</td>
<td>0.</td>
<td>994.094</td>
<td>309.263</td>
<td>325.500</td>
<td>1220.750</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>237</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>237</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>237</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td># WAITING</td>
<td></td>
<td>237</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>WAIT TIME</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>
| CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY
WAITING: NONE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT...</th>
<th>MEAN......</th>
<th>STD DEV...</th>
<th>MINIMUM...</th>
<th>MAXIMUM...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH1.B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.966</td>
<td>0.180</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>119</td>
<td>0.</td>
<td>0.034</td>
<td>0.180</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>119</td>
<td>0.</td>
<td>1015.532</td>
<td>203.448</td>
<td>813.750</td>
<td>1220.750</td>
</tr>
<tr>
<td># BUSY</td>
<td>119</td>
<td>0.</td>
<td>1015.532</td>
<td>203.448</td>
<td>813.750</td>
<td>1220.750</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>119</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>119</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>119</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td># WAITING</td>
<td></td>
<td>119</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>WAIT TIME</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>
CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY
WAITING: NONE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT...</th>
<th>MEAN...</th>
<th>STD DEV...</th>
<th>MINIMUM...</th>
<th>MAXIMUM...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH2.A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.935</td>
<td>0.247</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>237</td>
<td></td>
<td>0.065</td>
<td>0.247</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>237</td>
<td></td>
<td>994.092</td>
<td>309.202</td>
<td>326.500</td>
<td>1220.750</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>237</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>237</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
</tbody>
</table>
| CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY
WAITING: NONE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT...</th>
<th>MEAN...</th>
<th>STD DEV...</th>
<th>MINIMUM...</th>
<th>MAXIMUM...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH2.B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.965</td>
<td>0.180</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>119</td>
<td></td>
<td>0.034</td>
<td>0.180</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>119</td>
<td></td>
<td>1015.533</td>
<td>203.447</td>
<td>813.750</td>
<td>1220.750</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>119</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>119</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>RESOURCE</td>
<td>TOTAL NUMBER</td>
<td>CURRENT</td>
<td>MEAN ......</td>
<td>STD DEV ......</td>
<td>MINIMUM ......</td>
<td>MAXIMUM ......</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
<td>----------</td>
<td>-------------</td>
<td>----------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>CH3.A</td>
<td># IDLE</td>
<td>1.000</td>
<td>0.990</td>
<td>0.100</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>INTO BUSY</td>
<td>118</td>
<td>0.</td>
<td>0.010</td>
<td>0.100</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>OUT OF BUSY</td>
<td>118</td>
<td>0.</td>
<td>306.000</td>
<td>0.079</td>
<td>306.000</td>
</tr>
<tr>
<td></td>
<td>BUSY TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td># INACTIVE</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>INTO WAIT</td>
<td>118</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>OUT OF WAIT</td>
<td>118</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>CURRENTLY ALLOCATED TO PROCESSES: NONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PROCESSES CURRENTLY WAITING: NONE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH3.B</td>
<td># IDLE</td>
<td>1.000</td>
<td>1.000</td>
<td>0.</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>INTO BUSY</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>OUT OF BUSY</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>BUSY TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td># INACTIVE</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>INTO WAIT</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td></td>
<td>OUT OF WAIT</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>RESOURCE</td>
<td>TOTAL NUMBER</td>
<td>CURRENT...</td>
<td>MEAN...</td>
<td>STD DEV...</td>
<td>MINIMUM...</td>
<td>MAXIMUM...</td>
</tr>
<tr>
<td>----------</td>
<td>--------------</td>
<td>-------------</td>
<td>---------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>CH4.A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.907</td>
<td>0.112</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>238</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>238</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># BUSY</td>
<td></td>
<td>0.013</td>
<td>0.112</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>192.294</td>
<td>111.850</td>
<td>81.3/5</td>
<td>306.188</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>238</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>238</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># WAITING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAIT TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH4.B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.993</td>
<td>0.082</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># BUSY</td>
<td></td>
<td>0.007</td>
<td>0.082</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>203.448</td>
<td>0.037</td>
<td>203.375</td>
<td>203.500</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### RESOURCE: CH5.A

<table>
<thead>
<tr>
<th>TOTAL NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.999</td>
<td>0.033</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>48</td>
<td>0.</td>
<td>0.001</td>
<td>0.033</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>48</td>
<td>0.</td>
<td>0.081</td>
<td>0.078</td>
<td>0.250</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>0.</td>
<td>0.081</td>
<td>0.033</td>
<td>0.250</td>
<td>1.500</td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>48</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>48</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td># WAITING</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
</tbody>
</table>

### RESOURCE: CH5.B

<table>
<thead>
<tr>
<th>TOTAL NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.997</td>
<td>0.052</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>48</td>
<td>0.</td>
<td>0.003</td>
<td>0.052</td>
<td>1.000</td>
</tr>
<tr>
<td>OUT OF BUSY</td>
<td>48</td>
<td>0.</td>
<td>0.003</td>
<td>0.052</td>
<td>0.</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>0.</td>
<td>0.003</td>
<td>0.052</td>
<td>203.250</td>
<td>203.500</td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
</tbody>
</table>
### CH7.A

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.996</td>
<td>0.066</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>48</td>
<td>0.</td>
<td>0.004</td>
<td>0.066</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>325.615</td>
<td>0.051</td>
<td>325.600</td>
<td>325.750</td>
<td></td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO WAIT</td>
<td>48</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>OUT OF WAIT</td>
<td>48</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>

### CH7.B

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td># IDLE</td>
<td>1.000</td>
<td>0.949</td>
<td>0.220</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
<tr>
<td>INTO BUSY</td>
<td>166</td>
<td>0.</td>
<td>0.051</td>
<td>0.220</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td></td>
<td>1103.069</td>
<td>184.486</td>
<td>813.750</td>
<td>1220.750</td>
<td></td>
</tr>
</tbody>
</table>
PAGE 28

# INACTIVE 0. 0. 0. 0. 0.

INTO WAIT 186
OUT OF WAIT 186
# WAITING 0. 0. 0. 0. 1.000
WAIT TIME 0. 0. 0. 0.

CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

RESOURCE TOTAL
NUMBER CURRENT... MEAN...... STD DEV... MINIMUM... MAXIMUM...

CH0.A

# IDLE 1.000 0.962 0.191 0. 1.000
REQUEST TIME 788.898 1047.365 0. 4582.375

INTO BUSY 168
OUT OF BUSY 168
# BUSY 0. 0.038 0.191 0. 1.000
BUSY TIME 813.802 0.054 813.750 814.000

# INACTIVE 0. 0. 0. 0. 0.

INTO WAIT 168
OUT OF WAIT 168
# WAITING 0. 0.036 0.312 0. 0.000
WAIT TIME 788.898 1047.365 0. 4582.375

CURRENTLY ALLOCATED
TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

RESOURCE TOTAL
NUMBER CURRENT... MEAN...... STD DEV... MINIMUM... MAXIMUM...

CH0.B

# IDLE 1.000 0.945 0.228 0. 1.000
REQUEST TIME 0. 0. 0. 0.

INTO BUSY 286
OUT OF BUSY 286
# BUSY 0. 0.056 0.228 0. 1.000
BUSY TIME 694.874 440.708 325.500 1220.750
### RESOURCE

#### CH9.A

<table>
<thead>
<tr>
<th>Resource</th>
<th>Current</th>
<th>Mean</th>
<th>STD Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLE</td>
<td>1.000</td>
<td>1.000</td>
<td>0.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>BUSY</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

#### CH9.B

<table>
<thead>
<tr>
<th>Resource</th>
<th>Current</th>
<th>Mean</th>
<th>STD Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLE</td>
<td>1.000</td>
<td>1.000</td>
<td>0.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>REQUEST</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>BUSY</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>
PAGE 30
BUSY TIME 0. 0. 0. 0.
# INACTIVE 0. 0. 0. 0.
INTO WAIT 0
OUT OF WAIT 0
# WAITING 0. 0. 0. 0.
WAIT TIME 0. 0. 0. 0.
CURRENTLY ALLOCATED TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

RESOURCE TOTAL NUMBER CURRENT... MEAN... STD DEV... MINIMUM... MAXIMUM...
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
CHQ
# IDLE REQUEST TIME 1.000 0.726 0.446 0. 1.000
INTO BUSY 827
OUT OF BUSY 827
# BUSY BUSY TIME 1192.101 2058.180 0. 6000.000
# INACTIVE 0.
INTO WAIT 827
OUT OF WAIT 827
# WAITING WAIT TIME 0. 0. 0. 0. 0.
929.313 1924.185 0. 12589.750
CURRENTLY ALLOCATED TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

RESOURCE TOTAL NUMBER CURRENT... MEAN... STD DEV... MINIMUM... MAXIMUM...
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
DK1
# IDLE REQUEST TIME 1.000 0.998 0.646 0. 1.000
INTO BUSY 160
OUT OF BUSY 160
<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>HQ</td>
<td>410</td>
<td>1.000</td>
<td>0.728</td>
<td>0.445</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1576.327</td>
<td>2742.297</td>
<td>0.</td>
<td>15611.250</td>
<td></td>
</tr>
<tr>
<td></td>
<td>410</td>
<td>0.272</td>
<td>0.445</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2385.677</td>
<td>2616.605</td>
<td>0.</td>
<td>6000.063</td>
<td></td>
</tr>
<tr>
<td></td>
<td>410</td>
<td>0.122</td>
<td>0.380</td>
<td>0.</td>
<td>3.000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1073.888</td>
<td>2372.956</td>
<td>0.</td>
<td>15611.250</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>NUMBER</th>
<th>CURRENT</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>L3</td>
<td>96</td>
<td>1.000</td>
<td>0.925</td>
<td>0.263</td>
<td>0.</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>69.511</td>
<td>341.462</td>
<td>0.</td>
<td>2224.750</td>
<td></td>
</tr>
</tbody>
</table>

CURRENTLY ALLOCATED TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE
## SW1 Resource Statistics

<table>
<thead>
<tr>
<th>Resource</th>
<th>Current</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td># Idle</td>
<td>1.000</td>
<td>0.223</td>
<td>0.416</td>
<td>0.00</td>
<td>1.000</td>
</tr>
</tbody>
</table>

### Request Time

- Into Busy: 694
- Out of Busy: 694

### Busy Time

- Into Busy: 694
- Out of Busy: 694

### Inactive Time

- Into Wait: 694
- Out of Wait: 694

### Waiting Time

- Into Wait: 694
- Out of Wait: 694

## SW2 Resource Statistics

<table>
<thead>
<tr>
<th>Resource</th>
<th>Current</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td># Idle</td>
<td>1.000</td>
<td>0.729</td>
<td>0.445</td>
<td>0.00</td>
<td>1.000</td>
</tr>
</tbody>
</table>

### Request Time

- Into Busy: 694
- Out of Busy: 694

### Busy Time

- Into Busy: 694
- Out of Busy: 694

### Inactive Time

- Into Wait: 694
- Out of Wait: 694

### Waiting Time

- Into Wait: 694
- Out of Wait: 694
### PAGE 33

#### INTO BUSY

<table>
<thead>
<tr>
<th></th>
<th>214</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT OF BUSY</td>
<td>214</td>
</tr>
<tr>
<td>BUSY</td>
<td>0.</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>4664.486</td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.</td>
</tr>
</tbody>
</table>

#### INTO WAIT

<table>
<thead>
<tr>
<th></th>
<th>214</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT OF WAIT</td>
<td>214</td>
</tr>
<tr>
<td>WAITING</td>
<td>0.</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td>373.714</td>
</tr>
</tbody>
</table>

CURRENTLY ALLOCATED

TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE

#### RESOURCE

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>TOTAL NUMBER</th>
<th>CURRENT...</th>
<th>MEAN...</th>
<th>STD DEV...</th>
<th>MINIMUM...</th>
<th>MAXIMUM...</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># IDLE</td>
<td>1.00</td>
<td>0.542</td>
<td>0.498</td>
<td>0.</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>REQUEST TIME</td>
<td></td>
<td>3857.756</td>
<td>6603.183</td>
<td>0.</td>
<td>24434.500</td>
<td></td>
</tr>
</tbody>
</table>

#### INTO BUSY

<table>
<thead>
<tr>
<th></th>
<th>464</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT OF BUSY</td>
<td>464</td>
</tr>
<tr>
<td>BUSY</td>
<td>0.</td>
</tr>
<tr>
<td>BUSY TIME</td>
<td>3631.718</td>
</tr>
<tr>
<td># INACTIVE</td>
<td>0.</td>
</tr>
</tbody>
</table>

#### INTO WAIT

<table>
<thead>
<tr>
<th></th>
<th>464</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT OF WAIT</td>
<td>464</td>
</tr>
<tr>
<td>WAITING</td>
<td>0.</td>
</tr>
<tr>
<td>WAIT TIME</td>
<td>3857.756</td>
</tr>
</tbody>
</table>

CURRENTLY ALLOCATED

TO PROCESSES: NONE

PROCESSES CURRENTLY WAITING: NONE
## SIMULATION TIME = 3600000.0000 UNITS

### ACTION REPORT

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>100</td>
<td>2.000</td>
<td>0.</td>
<td>2.000</td>
<td>2.000</td>
<td>0.0005</td>
</tr>
<tr>
<td>DELAY</td>
<td>100</td>
<td>1523.546</td>
<td>2269.177</td>
<td>0.</td>
<td>9663.500</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>100</td>
<td>14.678</td>
<td>8.537</td>
<td>0.063</td>
<td>29.938</td>
<td>0.060</td>
</tr>
<tr>
<td>DELAY</td>
<td>100</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>2570</td>
<td>4337.743</td>
<td>1976.129</td>
<td>1599.992</td>
<td>6000.063</td>
<td>309.667</td>
</tr>
<tr>
<td>DELAY</td>
<td>2570</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>100</td>
<td>29.374</td>
<td>17.214</td>
<td>0.188</td>
<td>59.859</td>
<td>0.137</td>
</tr>
<tr>
<td>DELAY</td>
<td>100</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>354</td>
<td>7.522</td>
<td>6.272</td>
<td>0.</td>
<td>11.250</td>
<td>0.074</td>
</tr>
<tr>
<td>DELAY</td>
<td>354</td>
<td>1774.778</td>
<td>3092.568</td>
<td>0.</td>
<td>16811.260</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>100</td>
<td>0.019</td>
<td>0.037</td>
<td>0.</td>
<td>0.250</td>
<td>0.000</td>
</tr>
<tr>
<td>DELAY</td>
<td>100</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th>TOTAL SAMPLES</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>% OF TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEFUL</td>
<td>100</td>
<td>0.019</td>
<td>0.037</td>
<td>0.</td>
<td>0.250</td>
<td>0.000</td>
</tr>
<tr>
<td>DELAY</td>
<td>100</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td></td>
</tr>
</tbody>
</table>
SIMULATION TIME = 360000.0000 UNITS

PROCESS REPORT

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>TOTAL SAMPLES</th>
<th>SUM</th>
<th>MEAN</th>
<th>STD DEVIATION</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB-DATA</td>
<td>TOTAL</td>
<td></td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>PROCESS WAIT</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
<tr>
<td>RESOURCE WAIT</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
<td>0.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TOTAL</th>
<th># AUTO SCHEDULE</th>
<th># CALL SCHEDULE</th>
<th># NOT SCHEDULE</th>
<th>COMPLETE</th>
<th>COMPLETE SUSPEND</th>
</tr>
</thead>
<tbody>
<tr>
<td>118</td>
<td>118</td>
<td>0</td>
<td>118</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

PROCESS DESCRIPTION

AB-DATA
AIR BASE STATUS BROADCAST TO ALL OTHER NODES

<table>
<thead>
<tr>
<th>COUNT</th>
<th>ENTRY</th>
<th>Opcode</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>118</td>
<td>START</td>
<td>NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>RETURN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>CALL</td>
<td>MRS</td>
<td>NOWAIT</td>
<td>10</td>
<td></td>
<td>PROCESS REQUEST TO CHQ</td>
</tr>
<tr>
<td>118</td>
<td>GIVEN</td>
<td>CHQ-DATA</td>
<td>10</td>
<td>$REQNORE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>CALL</td>
<td>MRS</td>
<td>NOWAIT</td>
<td>10</td>
<td></td>
<td>PROCESS REQUEST TO HQ</td>
</tr>
<tr>
<td>118</td>
<td>GIVEN</td>
<td>HQ-DATA</td>
<td>10</td>
<td>$REQNORE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>ASSIGN</td>
<td>$NODE</td>
<td></td>
<td></td>
<td></td>
<td>CURRENT NODE</td>
</tr>
<tr>
<td>118</td>
<td>COMPARE</td>
<td>CNODE</td>
<td></td>
<td></td>
<td></td>
<td>TEST FOR CURRENT NODE</td>
</tr>
<tr>
<td>118</td>
<td>CALL</td>
<td>AB1</td>
<td>NOWAIT</td>
<td>10</td>
<td></td>
<td>PROCESS REQUEST TO AB1</td>
</tr>
<tr>
<td>59</td>
<td>GIVEN</td>
<td>ABUPDATE</td>
<td>10</td>
<td>$REQNORE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>BRANCH</td>
<td>END</td>
<td>100</td>
<td></td>
<td></td>
<td>BRANCH TO THE END</td>
</tr>
<tr>
<td>59</td>
<td>AB1</td>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td>ENTRY FROM COMPARE NODE</td>
</tr>
<tr>
<td>59</td>
<td>CALL</td>
<td>MRS</td>
<td>NOWAIT</td>
<td>10</td>
<td></td>
<td>PROCESS REQUEST TO AB2</td>
</tr>
<tr>
<td>59</td>
<td>GIVEN</td>
<td>ABUPDATE</td>
<td>10</td>
<td>$REQNORE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>END</td>
<td>ENTRY</td>
<td></td>
<td></td>
<td></td>
<td>ENTRY FROM REQUEST TO AB1</td>
</tr>
</tbody>
</table>

TOTAL
### Update Data From Airbase

<table>
<thead>
<tr>
<th>COUNT ENTRY</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>START</td>
<td>NO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>RETURN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>UPDATE</td>
<td>CONSTANT 0.1</td>
<td>TIME CONSUMED IN UPDATING</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PROCESS**

<table>
<thead>
<tr>
<th>TOTAL SAMPLES</th>
<th>SUM</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CHANPROC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS WAIT</td>
<td>2048</td>
<td>-7.940E+07</td>
<td>38798.344</td>
<td>33483.393</td>
<td>4813.750</td>
</tr>
<tr>
<td>RESOURCE WAIT</td>
<td>2048</td>
<td>129174.794</td>
<td>63.074</td>
<td>366.749</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TOTAL # AUTO</th>
<th># CALL</th>
<th># NOT</th>
<th># TIMES</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SCHEDULE</th>
<th>SCHEDULE COMPLETE</th>
<th>COMPLETE SUSPEND</th>
</tr>
</thead>
</table>

**ITEM**

<table>
<thead>
<tr>
<th>ITEM CREATED</th>
<th>RECEIVED</th>
<th>SENT</th>
<th>DESTR'D</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**PROCESS**

<table>
<thead>
<tr>
<th>PROCESS HOLDING TIME</th>
<th><strong>CHANPROC</strong></th>
<th><strong>FULL AND HALF DUPLEX CHANNEL LOGIC</strong></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>COUNT ENTRY</th>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>2048</td>
<td>START</td>
<td>ALL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>MSG</td>
<td>CNODE</td>
<td></td>
<td>SET INTERNAL NODE CURRENT</td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>$CNODE</td>
<td>TNODE</td>
<td></td>
<td>GET DESTINATION NODE</td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>MSG</td>
<td>TO.NODE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>$NXTNODE</td>
<td>TO.NODE</td>
<td></td>
<td>SET NEXT NODE TO DESTN</td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>$CHANNEL</td>
<td>TO.NODE</td>
<td></td>
<td>GET CHANNEL TO NEXT NODE</td>
</tr>
<tr>
<td>2048</td>
<td>ASSIGN</td>
<td>CHANNEL</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ALLOC CHANNEL 1 ALL OBTAIN CHANNEL FOR XFER
2048 ASSIGN CHANNEL RATE WHAT IS CHANNEL RATE?
2048 ASSIGN VSPEED LENGTH MESSAGE LENGTH
2048 ASSIGN VLENGTH MESSAGE LENGTH
2048 EVAL VM.DVHD MULTIPLY CALCULATE TRANSFER TIME
2048 XFER.OH CONSTANT VM.DVHD DELAY DUE TO TRANSFER TIME
2048 ASSIGN NXT.NODE MSG RESIDES IN NEXT NODE
2048 ASSIGN $CNODE SET INTERNAL NODE REGISTER
2048 DEALLOC CHANNEL 1 FREE UP CHANNEL AFTER XFER
2048 GIVEN MSG ROUTE MESSAGE TO NEXT NODE
2048 END

TOTAL SAMPLES SUM....... MEAN....... STD DEV... MINIMUM... MAXIMUM...
TOTAL 118 336577.578 2043.878 3260.058 11.250 12681.000
PROCESS WAIT 0 0. 0. 0. 0.
RESOURCE WAIT 103 334250.078 3245.146 3284.194 11.250 12689.750

TOTAL # AUTO # CALL # OF # NOT # TIMES
SCHEDULE SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND.
118 0 118 118 0 103

ITEM CREATED RECEIVED SENT DESTROYED
MSG 0 0 0 0 0

PROCESS HOLDING TIME
ITEM # SMPLS MEAN....... MINIMUM... MAXIMUM... STD DEV...
MSG 118 2043.88 11.26 12681.00 3260.00

PROCESS DESCRIPTION
CHQ-DATA CHQ GETS MESSAGE, FORMULATES RESPONSE, AND REPLIES
COUNT ENTRY OPCODE PARM PARM PARM COMMENT
110 START NO
110 GIVEN MSG
110 RETURN MSG
| 118 | ASSIGN MSG LENGTH              | MAKE MSG-LENGTH = V.LENGTH |
| 118 | EVAL V.LENGTH                  | EVALUATE MSG PROCESS TIME  |
| 118 | V.LENGTH                      |                           |
| 205 | UPDATE CONSTANT V.TIME        | PROCESSING TIME CONSUMED  |

**TOTAL**

<table>
<thead>
<tr>
<th>PROCESS SAMPLES</th>
<th>SUM</th>
<th>MEAN</th>
<th>STD DEV</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL 890</td>
<td>14897.168</td>
<td>25948.783</td>
<td>0.0</td>
<td>100000.125</td>
<td></td>
</tr>
<tr>
<td>PROCESS WAIT 690</td>
<td>14894.377</td>
<td>25102.148</td>
<td>0.0</td>
<td>100124.109</td>
<td></td>
</tr>
<tr>
<td>RESOURCE WAIT 522</td>
<td>798.792</td>
<td>2890.029</td>
<td>0.0</td>
<td>22000.008</td>
<td></td>
</tr>
</tbody>
</table>

**ITEM**

<table>
<thead>
<tr>
<th>ITEM</th>
<th>CREATED</th>
<th>RECEIVED</th>
<th>SENT</th>
<th>DESTROYED</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>0</td>
<td>0</td>
<td>690</td>
<td>0</td>
</tr>
</tbody>
</table>

**PROCESS**

<table>
<thead>
<tr>
<th>PROCESS HOLDING TIME</th>
<th># SMPLS</th>
<th>MEAN</th>
<th>MINIMUM</th>
<th>MAXIMUM</th>
<th>STD DEV</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG 1212</td>
<td>343.17</td>
<td>0.0</td>
<td>22000.01</td>
<td>1937.25</td>
<td></td>
</tr>
</tbody>
</table>

**DESTPROC**

<table>
<thead>
<tr>
<th>COUNT ENTRY</th>
<th>_OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>690</td>
<td>START</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>690</td>
<td>GIVEN</td>
<td>MSG</td>
<td>CNODE</td>
<td></td>
<td>CURRENT NODE</td>
</tr>
<tr>
<td>690</td>
<td>ASSIGN</td>
<td>MSG</td>
<td>C.NODE</td>
<td></td>
<td>IF RESPONSE, DESTROY</td>
</tr>
<tr>
<td>690</td>
<td>COMPARE</td>
<td>MSG</td>
<td>TYPE</td>
<td>EQ</td>
<td>EXECUTE THE CALLED PROCESS</td>
</tr>
<tr>
<td>522</td>
<td>ALLOC</td>
<td>C.NODE</td>
<td>1</td>
<td>ALL</td>
<td></td>
</tr>
<tr>
<td>522</td>
<td>ASSIGN</td>
<td>MSG</td>
<td>RPROC</td>
<td></td>
<td>SET PRIORITY FOR REQ PROC</td>
</tr>
<tr>
<td>522</td>
<td>ASSIGN</td>
<td>MSG</td>
<td>RPROC</td>
<td>PRI</td>
<td></td>
</tr>
<tr>
<td>522</td>
<td>CALL</td>
<td>PROCESS</td>
<td>WAIT</td>
<td>PRIOR</td>
<td>UNTIL COMPLETE</td>
</tr>
</tbody>
</table>
RETURN MSG
DEALLOC C.NODE 1 TYPE EQ DEALLOCATE CURRENT NODE
COMPARE MSG $REQNODE DESTROY NO RESPONSE REQ -> DESTROY
ASSIGN $RESP MSG TYPE CHANGE MSG RESPONSE TYPE
ASSIGN MSG FNODE SWITCH FROM AND TO NODES
ASSIGN MSG TNODE CURRENT NODE IS FROM NODE
CALL CHANPROC WAIT 0 RETURN MESSAGE TO ORIGIN
GIVEN MSG BRANCH END 100 TERMINATE MESSAGE AT DEST
DESTROY ENTRY
DESTROY MSG TERMINATE MSG
END

PROCESS TOTAL SAMPLES SUM........MEAN......STD DEV...MINIMUM...MAXIMUM
===================================------------------------------------------
DISK.OP
TOTAL PROCESS WAIT 168 7403.990 44.071 18.937 3.656 85.600
PROCESS WAIT 168 0 0 0 0 0 0

TOTAL # # AUTO # CALL # OF # NOT # TIMES
SCHEDULE SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND.
=================================------------------------------------------
168 0 168 168 0 0 0

PROCESS DESCRIPTION
=================================------------------------------------------
DISK.OP OPERATION OF DISK
COUNT ENTRY OPCODE PARM PARM PARM COMMENT
=========== =========== =========== =========== ===========
168 188 188 188 188 188 START GIVEN ASSIGN EVAL ALLOC ASSIGN SEEK ASSIGN
LENGTH DISK SPEED EXTME V.SPEED DISK SEEK SEEK TIME
NO
MAKE DISK SPEED = V.SPEED
TRANSFER TIME CALCULATED
PARTIAL DISK ALLOCATED
MAKE SEEKTIME = SEEK
TIME FOR SEEK IS CONSUMED
MAKE DISK LATENCY=LATETIME
LATETIME
LATENCY
UNIFORM
LATETIME
LATE TIME
TIME CONSUMED FOR LATENCY
XFER
CONSTANT
XFER TIME
TRANSFER TIME CONSUMED
DEALLOC
DISK
1
DISK RESOURCE DEALLOCATED

TOTAL
SAMPLES.
SUM.
MEAN.
STD DEV.
MINIMUM.
MAXIMUM.

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

HQ-DATA
TOTAL
PROCESS WAIT
RESOURCES
WAIT

TOTAL
# AUTO
# CALL
# SCHEDULE
# SCHEDULE
# COMPLETE
# COMPLETE
# SUSPEND.

ITEM
CREATED
RECEIVED
SENT
DESTR'D

MSG
0
0
0
78

ITEM
# SCHEDULE
# COMPLETE
# SUSPEND.

MSG
118
2090.90
11.25
16022.60
3636.78

PROCESS
DESCRIPTION

HQ-DATA
HQ GETS MESSAGE, FORMULATES RESPONSE, AND REPLIES

COUNT
ENTRY
OPCODE
PARM
PARM
PARM
PARM
COMMENT

118
START
NO

118
GIVEN
MSG

118
ASSIGN
MSG
LENGTH
MAKE MSG-LENGTH = V.LENGTH

118
EVAL
V.TIME
MULTIPLY
EVALUATE MSG PROCESS TIME

118
UPDATE
V.TIME
PROCESSING TIME CONSUMED

TOTAL
SAMPLES.
SUM.
MEAN.
STD DEV.
MINIMUM.
MAXIMUM.

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

HQ-REQ
TOTAL
48
+1.852E+06
38585.016
9496.574
27754.250
63392.188
## PROCESS WAIT

<table>
<thead>
<tr>
<th>OPCODE</th>
<th>PARM</th>
<th>PARM</th>
<th>PARM</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>L3</td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GIVEN</td>
<td>MSG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>WRS</td>
<td>WAIT</td>
<td>4</td>
<td>MAKES I/O REQUEST TO CHQ</td>
</tr>
<tr>
<td>GIVEN</td>
<td>PLANS</td>
<td>4</td>
<td>$REQRESP</td>
<td></td>
</tr>
<tr>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## PROCESS DESCRIPTION

```plaintext
HQ-REQ
HQ REQUEST FOR STATUS DISPLAY FROM CHQ
```

## RESOURCE WAIT

```plaintext
TOTAL # AUTO CALL OF NOT TIMES
SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND.

<table>
<thead>
<tr>
<th>ITEM</th>
<th>CREATED</th>
<th>RECEIVED</th>
<th>SENT</th>
<th>DESTR'D</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG</td>
<td>522</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

## PROCESS HOLDING TIME

### ITEM COUNT(S) SUM MEAN MINIMUM MAXIMUM STD DEV

```
```
ASSIGN WSG C.NODE  
ASSIGN C.NODE W.ROUTE  
ASSIGN WSG LNGTH  
EVAL OVERHEAD MULTIPLY  
ALLOC C.NODE 1 ALL  
ROUTE.OH CONSTANT OVERHEAD  
DEALLOC C.NODE 1  
COMPARE WSG C.NODE EQ  
CALL CHANPROC WAIT 0  
GIVEN WSG  
BRANCH END 100  
CONTROL ENTRY  
CALL DESTPROC WAIT 0  
GIVEN WSG  
END ENTRY  
END

PROCESS SAMPLES SUM ........ MEAN ........ STD DEV .... MINIMUM .... MAXIMUM .... 

PLANS

TOTAL  168  270504.811  1610.148  2264.766  9.000  9723.125  
PROCESS WAIT  168  7403.990  44.071  18.937  3.650  86.500  
RESOURCE WAIT  152  252764.020  1728.716  1646.363  11.250  8180.375  

TOTAL AUTO CALL OF NOT TIMES 
SCHEDULE SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND.  
168 0 168 168 0 0 152

ITEM CREATED RECEIVED SENT DISTR  
MSG 0 0 0 0 0

ITEM # SMPLS MEAN .... MINIMUM .... MAXIMUM .... STD DEV .... 

PROCESS HOLDING TIME  
MSG  168  1610.15  9.00  9723.13  2264.76

PROCESS DESCRIPTION  
PLANS REQUEST FOR PLANS FROM CHQ
START CHQ NO
GIVEN MSG
RETURN MSG
ASSIGN MSG LENGTH MAKE MSG LENGTH = V.LENGTH
EVAL V.TIME MULTIPLY EVALUATE MSG PROCESS TIME
01 V.LENGTH
FORMAT CONSTANT V.TIME TIME USED TO FORMAT PLANS
CALL DISK.OP WAIT 10 CALLING PROCESS DISK.OP
GIVEN 500 DK1
ASSIGN 500 MSG LENGTH INCREASE MSG LENGTH
END

TOTAL SAMPLES SUM..... MEAN..... STD DEV... MINIMUM... MAXIMUM...
TRACETOTAL 1 0. 0. 0. 0. 0.
PROCESS WAIT 0 0. 0. 0. 0. 0.
RESOURCE WAIT 0 0. 0. 0. 0. 0.
TOTAL # AUTO # CALL # OF SCHEDULE SCHEDULE COMPLETE COMPLETE SUSPEND.
SCHEDULE COMPLETE COMPLETE SUSPEND.
1 1 0 1 0 0

PROCESS DESCRIPTION
TRACETURN ON TRACE OUTPUT
COUNT ENTRY OPCODE PARM PARM PARM COMMENT
START ALL NO
COMPARE V.TRACE EQ TEST IF FLAG SET FOR TRACE
TRACENO ON
NOTRACE ENTRY
END