Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 73.
# Figures

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>The Configuration Definition screen</td>
<td>. . . . . . . . . . . 9</td>
<td>16.</td>
<td>Model input - the LPAR Definition screen</td>
<td>. . . . . . . . . . . 36</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>The Workload Definition screen</td>
<td>. . . . . . . . . . . 11</td>
<td>17.</td>
<td>The Calibration Form screen</td>
<td>. . . . . . . . . . . 38</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>The LPAR Definition screen</td>
<td>. . . . . . . . . . . 13</td>
<td>18.</td>
<td>The RMF Extraction screen</td>
<td>. . . . . . . . . . . 42</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>The Run Wizard screen</td>
<td>. . . . . . . . . . . 16</td>
<td>19.</td>
<td>The Model Build form</td>
<td>. . . . . . . . . . . 45</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>The Multi Image Model Creation screen</td>
<td>. . . . . . . . . . . 18</td>
<td>20.</td>
<td>The CPU report</td>
<td>. . . . . . . . . . . 47</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>The Simulator Running screen</td>
<td>. . . . . . . . . . . 20</td>
<td>21.</td>
<td>The Workload Selection screen</td>
<td>. . . . . . . . . . . 48</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>The Graph Selection screen</td>
<td>. . . . . . . . . . . 22</td>
<td>22.</td>
<td>Simulation Results summary</td>
<td>. . . . . . . . . . . 54</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>The Performance Tracking screen</td>
<td>. . . . . . . . . . . 23</td>
<td>23.</td>
<td>Lotus 1-2-3 spreadsheet file</td>
<td>. . . . . . . . . . . 62</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>The Processor Utilization Tracking screen</td>
<td>. . . . . . . . . . . 24</td>
<td>24.</td>
<td>Imported data</td>
<td>. . . . . . . . . . . 63</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>The Print Option screen</td>
<td>. . . . . . . . . . . 25</td>
<td>25.</td>
<td>The CPU utilization graph</td>
<td>. . . . . . . . . . . 64</td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>A workload activity report</td>
<td>. . . . . . . . . . . 30</td>
<td>26.</td>
<td>Imported workload figures</td>
<td>. . . . . . . . . . . 65</td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Workload analysis</td>
<td>. . . . . . . . . . . 31</td>
<td>27.</td>
<td>Workload Utilization graph</td>
<td>. . . . . . . . . . . 66</td>
<td></td>
</tr>
<tr>
<td>13.</td>
<td>Simulator input</td>
<td>. . . . . . . . . . . 32</td>
<td>28.</td>
<td>Workload Analysis Table</td>
<td>. . . . . . . . . . . 68</td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Model input - the Configuration Definition screen</td>
<td>. . . . . . . . . . . 34</td>
<td>29.</td>
<td>The SIMN sheet</td>
<td>. . . . . . . . . . . 70</td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>Model input - the Workloads Definition screen</td>
<td>. . . . . . . . . . . 35</td>
<td>30.</td>
<td>Summarizing results of many runs</td>
<td>. . . . . . . . . . . 71</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 1. Introduction and Overview

Introduction

Performance Modeler is a performance modeling tool which runs under the Windows® 98 (or higher level) operating system on a personal computer.

Performance Modeler can model performance characteristics at the individual workload level for MVS™, OS/390®, or z/OS™ based mainframes.

Performance Modeler is a Systems Management tool. It is designed to help Information Technology shops manage their day to day performance as well as their long term Capacity Planning functions. This manual is intended for the IT professional who has experience with performance analysis and capacity planning.

Performance Modeler falls into the class of tools described as models. Models are tools that are easy to use, yet can predict the output of complex systems. Models are valuable because they let you examine a number of “what if” scenarios without the expense of running the actual scenarios on the system being studied. In the world of S/390® mainframes, predicting the impact of changing hardware or software can be a difficult task.

Performance Modeler uses sophisticated simulation techniques to model the performance of these systems with a high degree of accuracy. One of the advantages of Performance Modeler is that it is a PC-based tool. That means it can be run over and over without using mainframe resources. Plus, Performance Modeler is a portable tool, and can be run wherever your PC or laptop is located.

The only changes between this edition of the User’s Guide and the previous edition is that some small bugs have been corrected. These changes are reflected in some screen captures. They are tagged with a vertical revision bar.

Simulation versus analytic models

Modeling techniques generally fall into one of two categories, simulation or analytic. In some cases, hybrid models have been developed which use combinations of both techniques. The differences between these techniques are:

• Analytic models consist of mathematical equations which describe the processes being modeled. Analytic models are used when the processes being modeled are well understood and can be described as mathematical expressions.

Simulation models rely on running the actual process being modeled, but in a simplified form. Since these processes take place over time, simulation requires running (simulating) these processes over and over in order to reach a steady state that mimics the real process.

• Analytic models are by their nature fast to run and require less processing power than simulation. Analytic models execute a number of mathematical equations and can run in seconds or less.

Computer based simulators must execute lots of instructions in order to simulate one point in time. And since simulators must model a period of time, lots of iterations must take place before the results are meaningful. This means simulators can run for an extended amount of real time, and use a larger amount of computer resources compared to analytic models.
Simulation versus analytic models

- Simulators are used today to predict complex processes where analytic equations are not suitable. Some examples of these simulators are models which are used to predict weather and models which predict the presence of oil deposits. These models require large amounts of computer resources but produce results that are not attainable with analytic techniques.

In the early days of mainframe computers, operating systems were simpler than they are today. But over time, features have been added and operating system complexity has grown. Older analytic models which were developed to predict computer performance have found it difficult to keep up with the changing nature and complexity of today’s systems.

Simulation models have a distinct advantage over analytic models when it comes to staying current. Simulators can be easily reprogrammed to model changes in the operating system as they occur.

These differences between analytic and simulation models are some of the reasons why Performance Modeler was developed as a simulation model. Although Performance Modeler uses simulation techniques, it is very efficient. This means the time it takes to reach a set of meaningful results is quite fast. In fact, running on modern PCs, Performance Modeler runs in less real time than the amount of time actually being simulated.

Model input and output

Performance Modeler can model the changes to nearly every hardware and software change that might be made. This makes Performance Modeler quite powerful in being able to answer “what if” scenarios.

For example, Performance Modeler can model the effects of hardware configuration changes such as the number of CPUs, the speed of the CPUs, Disk I/O response times, and paging rates (auxiliary and Expanded Memory paging). Performance Modeler also models the effects of changing LPAR (Logical Partitioning) parameters, including the number of logical CPUs per LPAR and their Weighting Factors. You can also change the amount of work being run to model the impact of increased or decreased workload volumes.

All of these inputs are specified on three easy-to-understand screens. These are the Configuration screen, the Workload Activity screen, and the LPAR Definition screen.

The output from running Performance Modeler includes standard CPU utilization reports for the entire processor, as well as a breakdown of utilization by individual LPAR. But the most important output metric is the performance of each workload being modeled. For online workloads, this is average response time (in seconds). For batch workloads, this is average elapsed time factors (this is described in more detail in Chapter 5, “Running the simulator,” on page 15).

The ability to model the performance of individual workloads makes Performance Modeler a powerful tool for Performance Management and Capacity Planning.

Workload types and simulation

All workloads are defined to Performance Modeler as one of three different workload types. These are single tasking workloads (Type=S), multi-tasking workloads (Type=M), and batch workloads (Type=B).
Type S and Type M are both used for defining online workloads. Online workloads have these attributes:

- They are made up of transactions that enter the system in a random pattern and require a relatively short burst of processing capacity.
- Each transaction executes independently from other transactions.

CICS® regions and IMS™ message processing regions are examples of online workloads. But so are TSO generated transactions.

Single tasking workloads are made up of transactions which run under a single TCB (Task Control Block) or single task. These transactions can only run on one CPU at a time. Multi-tasking workloads represent workloads whose transactions can execute on multiple CPUs at the same time. A single CICS region is an example of a workload that should be defined as single tasking. Multi-tasking workloads can represent a number of TSO users, or they can represent a group of workloads that are defined as one consolidated workload.

Two important fields are used to define online workloads. These are Average Arrival Rate and Average Path Length. Average Arrival Rate is the average transaction rate (transactions per second) and determines the rate at which new transactions enter the system. The Average Path Length defines the average number of instructions per transaction.

When these two parameters are multiplied, the result is the average MIPS consumed by this workload. For example, [transactions per second] X [# of instructions in Millions per transaction] = Millions of instructions per second or MIPS consumed. This is a handy relationship to remember since it is an easy check to see how much capacity each online workload tries to consume.

Batch workloads (Type=B) are handled and modeled differently from online workloads. Batch jobs appear to the model as transactions that never end. Each batch job executes a number of instructions, stops to perform an I/O operation, then resumes executing again when the I/O completes. Unlike online workloads, there is no Arrival Rate to define. Batch jobs are either executing, waiting to execute, or performing I/O operations. For batch workloads, the Average Path Length represents the average # of instructions that must execute before stopping for a synchronous I/O operation. The ratio of Average Path Length to the time required to perform an I/O operation determines whether the batch job is CPU bound or I/O bound. This ratio is also one of the key factors in determining how many MIPS are consumed by this workload. The more CPU bound the job, the greater the MIPS the job tries to consume.

### Simulation techniques

When the model runs, real time is divided into increments of simulated time. At the beginning of each time interval, the model examines each workload. For online workloads, the model determines if it is time to generate a new transaction. The model generates new transactions based on the Average Arrival Rate. Batch jobs are represented as never ending transactions so they are always present.

Both batch and online workloads may be suspended due to paging activity or when higher priority workloads preempt their execution. More information on how the model works is included in Chapter 6, “Getting started - performance modeling basics,” on page 27.
Chapter 2. Performance Modeler installation and system requirements

System requirements

Performance Modeler is a Microsoft® Windows-based program. Performance Modeler runs on any PC capable of running Windows 98 (or later release). Due to the CPU intensive nature of simulation, a PC internal clock speed of 300 MHz or greater is recommended. Performance Modeler supports different screen display sizes, but 1024 by 768 is recommended for optimum viewing.

This program has been developed and tested in a Windows environment with the locale set to English-United States (EN-US). The program, and the Lotus 123 and Excel scripts, use decimal notation with a period as the decimal symbol, for example “123.4”. Users of the program will have to ensure that the locale is set to EN-US or other compatible locale.

Installation instructions

Here is how to install Performance Modeler:
1. Run Setupwin32.exe from the root directory on the distribution CD-ROM. The Welcome screen is displayed.
2. Press Next to continue with the installation. Pressing Cancel will cancel the installation. The Software License Agreement is displayed.
3. Please read the license agreement. Select “I accept the terms in the license agreement” to continue with the installation. The Directory Name screen is displayed.
   If you select “I do not accept the terms in the license agreement”, then the Decline License screen is displayed. If you press Yes, then the installation is cancelled, and if you press No, then the Software License Agreement is displayed again.
4. If you wish to change the folder to which Performance Modeler is installed, do so now.
   To move to the next screen, press Next. The confirmation screen is displayed.
5. Press Next to continue the installation.
   The Installing screen is displayed, with a progress bar showing the state of the installation.
6. Press Finish to complete the process.

To launch Performance Modeler select Start/Programs/IBM Tivoli Performance Modeler/IBM Tivoli Performance Modeler.

Uninstall Performance Modeler using Start/Programs/IBM Tivoli Performance Modeler/Uninstall or the standard Start/Settings/Control Panel/Add/Remove Programs method.

Following a successful installation, you should have these files in the folder you selected for installation:
Installation instructions

<table>
<thead>
<tr>
<th>Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPFPM.EXE</td>
<td>Executable program</td>
</tr>
<tr>
<td>CPUMIPS.DAT</td>
<td>Table of processor MIPS ratings</td>
</tr>
<tr>
<td></td>
<td>The latest CPUMIPS.DAT file with all current processor performance information can be found at <a href="http://www-3.ibm.com/software/sysmgmt/products/support/IBMTivoliPerformanceModelerforzOS.html">http://www-3.ibm.com/software/sysmgmt/products/support/IBMTivoliPerformanceModelerforzOS.html</a> under Support Flashes.</td>
</tr>
<tr>
<td>DEFAULT.SIM</td>
<td>Default model definition statements</td>
</tr>
<tr>
<td>CPFPM.HLP</td>
<td>Help file</td>
</tr>
<tr>
<td>CPFUGA00n.PDF</td>
<td>User’s Guide</td>
</tr>
<tr>
<td>CPFPM.123</td>
<td>123 spreadsheet</td>
</tr>
<tr>
<td>CPFPM.XLS</td>
<td>EXCEL spreadsheet</td>
</tr>
</tbody>
</table>

You should also have the following subdirectories:

<table>
<thead>
<tr>
<th>Subdirectory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_uninst</td>
<td>Product uninstall information</td>
</tr>
<tr>
<td>license</td>
<td>License text files in national languages</td>
</tr>
</tbody>
</table>
Chapter 3. The Primary menu

The Primary menu is the launching point for all Performance Modeler operations. The Primary menu provides options for entering model definition parameters as well as running the simulator. The following section describes each of these options.

<table>
<thead>
<tr>
<th>Menu option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>The File menu lets you save the current model or load a previously saved model. All models should be saved with the SIM file type. If no file type is specified, the SIM type is automatically added to the file name.</td>
</tr>
<tr>
<td>Run</td>
<td>The Run menu specifies the type of simulation to be performed. Five types of simulations are possible. These are Single Run, Multiple Run, Calibration Run, Run Wizard, and Multi Image Run. These options are described more fully in Chapter 5, “Running the simulator,” on page 15.</td>
</tr>
<tr>
<td>Configuration</td>
<td>The Configuration option displays the Configuration Definition screen. This is screen 1 of 3 input panels. The Configuration Definition screen lets you choose the configuration to be modeled. Detailed information on these input fields is shown in Chapter 4, “Input screens,” on page 9.</td>
</tr>
<tr>
<td>Workloads</td>
<td>The Workloads option displays the Workload Definition screen. This is screen 2 of 3 input panels. This is where you define the workloads to be modeled. Detailed information on these input fields is shown in Chapter 4, “Input screens,” on page 9.</td>
</tr>
<tr>
<td>LPAR</td>
<td>The LPAR option displays the LPAR Definition screen. This is screen 3 of 3 input panels. The LPAR Definition Screen is where you specify information about LPAR mode. Detailed information on these input fields is shown in Chapter 4, “Input screens,” on page 9.</td>
</tr>
<tr>
<td>Edit</td>
<td>The Edit option provides a simple way to move workload and LPAR definitions. Individual Workload or LPAR definitions can be cut, copied, and pasted within the same configuration file, or to other configuration files. This provides a simple point and click facility for moving work between different models. This capability can be used when modeling the impact of moving work within a SYSPLEX.</td>
</tr>
<tr>
<td>Extract</td>
<td>The Extract option provides a data reduction program which reads RMF™ reports and extracts key performance metrics. The extracted data is stored in a format that can be easily imported into spreadsheet programs. The extracted data is also used by the Build option to automatically build a new model. The Extract option is described in Chapter 7, “Extracting RMF report data,” on page 41.</td>
</tr>
<tr>
<td>Build</td>
<td>The Build option provides the ability to automatically generate the model input parameters. This function is described in Chapter 8, “Automatically generating a model with the Build option,” on page 45.</td>
</tr>
<tr>
<td>Merge</td>
<td>The Merge option lets you combine several text files together into a</td>
</tr>
</tbody>
</table>
single text file. This may be used to combine several RMF/CMF report files into a single file. Later chapters show how to extract reporting information from these report files.

**View**

The View option lets you toggle on or off the Expanded Tab Feature. This feature helps people with vision problems, or other disabilities that stop them using a mouse. In Expanded Tab mode, you can change the cursor position so that it points to labels and other descriptive fields on a screen. In the default mode (Expanded Tab disabled), pressing the tab key only moves the cursor to editable input fields, or to command buttons.
Chapter 4. Input screens

Input to Performance Modeler can be hand entered or it can be generated using an automated BUILD function. The BUILD function is described in Chapter 8, "Automatically generating a model with the Build option," on page 45. This chapter describes the input fields contained in the three input screens. Techniques for determining these values are covered in later chapters.

The Configuration Definition screen

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE FOR THIS RUN</td>
<td>Default Model</td>
</tr>
<tr>
<td>TOTAL RUN TIME (sec)</td>
<td>200</td>
</tr>
<tr>
<td>TIME INTERVAL (sec)</td>
<td>0.0100</td>
</tr>
<tr>
<td># OF SECONDS PER REPORT</td>
<td>6</td>
</tr>
<tr>
<td>CPU DESCRIPTION</td>
<td>9672-R56</td>
</tr>
<tr>
<td>CPU SPEED (mips)</td>
<td>52.72</td>
</tr>
<tr>
<td># OF CPUs (1-32)</td>
<td>6</td>
</tr>
<tr>
<td>PAGE PACK RESP. TIME (sec)</td>
<td>0.020</td>
</tr>
<tr>
<td>E-STOR. RESP. TIME (sec)</td>
<td>0.000030</td>
</tr>
<tr>
<td># OF WORKLOADS (1-20)</td>
<td>7</td>
</tr>
<tr>
<td># OF LPARS (0-15)</td>
<td>2</td>
</tr>
</tbody>
</table>

This panel contains information about the processor and configuration being modeled. Here is a description of each input field.

**Field Name**  **Field Description**

**Title For This Run**
A description of this model (documentation only).

**Total Run Time (sec)**
The total amount of time in seconds that is simulated. A default of 200 seconds is suitable for most models.
The Configuration Definition screen

**Time Interval (sec)**

The simulator breaks up real time into discrete intervals based on this parameter. The smaller the interval, the more accurate the simulation. But a smaller interval also increases the time required to complete the simulation. The default interval of .01 seconds means the simulator performs 100 simulation passes in each second of simulated time.

**# Of Seconds Per Report**

The simulator updates the results in the Running screen after this number of simulated seconds have elapsed. For example, a value of 5 means the results are updated after every 5 seconds of simulated time.

**CPU Description**

Description of CPU being modeled (documentation only).

**CPU Speed (mips)**

MIPS rating (Millions of Instructions Per Second) for the CPU being modeled. This is the MIPS rating for each engine, not the entire processor. For example, if the processor is a 5 way MP with a total MIPS rating of 100 then the MIPS rating for a single engine would be 20 MIPS. The MIPS rating is a measure of processor capacity.

**# Of CPUs (1-32)**

The # of CPUs being modeled. For a 5 way MP, the # would be 5.

**Page Pack Resp. Time (sec)**

The average response time (seconds) to read a page (4K block) from disk storage.

**E-Stor Resp. Time (sec)**

The average time to move a page (4K block of memory) between Expanded Storage and Central Storage.

**# Of Workloads (1-20)**

The # of workloads that are modeled.

**# Of LPARS (0-15)**

The # of LPARS to be modeled. A value of 0 means no LPAR Modeling.
The Workload Definition screen

Figure 2 is an example of the Workload Definition screen.

This panel contains information about each workload being modeled. Up to 20 workloads can be defined.

The D and R buttons to the left of each workload provide a fast way to delete (D) or replicate (R) this workload definition. The S button is used to select the workload for editing. Once selected, the workload definition fields will change color indicating it is selected. You can now click on the Edit menu function where you can perform a Cut, Copy, or Insert operation. The Copy operation copies the selected workload definitions to an internal clipboard. The Cut operation is identical to Copy but also removes the selected workload from the active model. The Insert operation pastes the contents from the internal clipboard immediately after the selected workload. These functions provide a simple way to move workloads from one model to another without the need to reenter workload definitions. The S button is also available on the LPAR Definition screen and allows you to move LPAR definitions in a similar way.

The X button to the left of the Arrival Rate provides a simple way to multiply the Arrival Rate. Use this button when you want to increase or decrease the size of the workload. For example, selecting the X button and entering a multiplication factor of 2 doubles the Arrival Rate for this workload.
The Workload Definition screen

Here is a description of each input field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workload</td>
<td>Description of workload being modeled (documentation only).</td>
</tr>
<tr>
<td>Prty</td>
<td>Dispatching priority for this workload. Highest priority being 1, lowest being 99.</td>
</tr>
<tr>
<td>Path Length (X 1M)</td>
<td>Average # of instructions for a transaction (for online workloads). For batch workloads, this is the average # of instructions between I/O Operations.</td>
</tr>
<tr>
<td>Arrival Rate</td>
<td>Average transaction rate (transactions per second) for this workload. For batch workloads, this value should be 0.</td>
</tr>
<tr>
<td>I/O Rate</td>
<td>Average # of I/O operations per online transaction. For batch workloads, this value should be 0.</td>
</tr>
<tr>
<td>Page Fault Rate</td>
<td>Average # of page faults per second.</td>
</tr>
<tr>
<td>Disk Resp.</td>
<td>Average response time (seconds) to perform a disk I/O operation.</td>
</tr>
<tr>
<td>E-Stor Rate</td>
<td>Average # of pages per second that move between Expanded Storage and Central Storage.</td>
</tr>
<tr>
<td># of Users</td>
<td>For online workloads (Type=M or S), # of Users represents the # of online users that generate transactions for this workload. If set to 0, transactions are generated based on the arrival rate. If set greater than 0, transactions are generated based on arrival rate until the # of queued transactions equals the # of Users. Once the queue depth reaches the # of Users, no new transactions are generated until the queue depth drops below the # of Users. For Batch (Type=B) workloads, # of Users represents the multi-programming level (MPL). This is the # of identical batch jobs that make up this workload.</td>
</tr>
<tr>
<td>Trans Type (S/M/B/#)</td>
<td>Trans Type refers to what kind of workload to generate. Type=S is an online, single tasking workload which can only run on one engine at a time. Type=M is an online, multi-tasking workload. Multi-tasking workloads can execute on multiple engines concurrently. Type=B is a batch workload. Each Batch job can only run on one CPU, but Batch workloads with a multi-programming level (MPL) greater than one can spread individual jobs across multiple CPUs. Use Type=# to generate multiple transactions or applications which run in a single region as a single task. For example, if 2 workloads are defined as Type=3, then both workloads run under the same task but are defined and tracked as separate workloads. Use this option to define multiple CICS transactions that run in the same region under the same TCB.</td>
</tr>
</tbody>
</table>

The LPAR Definition screen

Figure is an example of the LPAR Definition screen.
This panel contains the LPAR (Logical Partition) definitions for the model.

Here is a description of each input field.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPAR Overhead (%)</td>
<td>The % overhead due to running LPAR mode. This represents the % of capacity lost to running LPAR Mode.</td>
</tr>
<tr>
<td># LCPUs</td>
<td># of Logical CPUs allocated to this LPAR.</td>
</tr>
<tr>
<td>Weighting Factor</td>
<td>The Weighting Factor for this LPAR.</td>
</tr>
<tr>
<td>MIPS</td>
<td>The total amount of MIPS that represent the workloads in this LPAR. MIPS can only be entered for LPARS # 2 thru 15. The model always runs as LPAR # 1. The MIPS consumed by the model are dynamically calculated by the model as part of the simulation. The MIPS consumed by LPARS # 2 thru 15 are input data and represent the amount of capacity that is required to service these LPARs.</td>
</tr>
<tr>
<td>Capping Yes No</td>
<td>Specifies whether LPAR Capping is to be enforced. When an LPAR is Capped, it is limited to using a fraction of the total capacity of the processor. The fraction it can use is determined by the Weighting Factor. The fraction is calculated as Weight (this LPAR)</td>
</tr>
</tbody>
</table>
divided by the sum of the Weights for all LPARs. Under normal operation (No Capping), LPAR weights are only enforced when the processor utilization is at or near 100% busy. But when Capping is turned on, the LPAR weight is always enforced.
## Chapter 5. Running the simulator

### Run options

The simulation process begins by selecting one of the five Run pull down menu options. These options include Single Run, Multiple Run, Calibration Run, Run Wizard, and Multi Image Run. The following section describes each of these options.

<table>
<thead>
<tr>
<th>Run Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Run</td>
<td>Selecting Single Run starts the simulation process using the currently loaded model.</td>
</tr>
<tr>
<td></td>
<td>Unless you stop the simulator by clicking the Stop button, the model continues to run until is has simulated the Total Run Time you selected. The Total Run Time is specified in the Configuration Definition screen.</td>
</tr>
<tr>
<td></td>
<td>At the completion of the run, you can select one of these options:</td>
</tr>
<tr>
<td>Restart</td>
<td>Restart the model from the beginning</td>
</tr>
<tr>
<td>Return</td>
<td>Return to the Input panels</td>
</tr>
<tr>
<td>Print</td>
<td>Print the results of the model</td>
</tr>
<tr>
<td>Multiple Run</td>
<td>Multiple Run lets you run multiple models in a batch mode.</td>
</tr>
<tr>
<td></td>
<td>When you select Multiple Run, Performance Modeler displays a panel that asks you to specify up to 15 model file names. These are the names of the models which have been previously saved on disk.</td>
</tr>
<tr>
<td></td>
<td>Each model should be saved with the file type SIM. If the file name is entered without an extension, a SIM extension is automatically added. After each model completes its run, the results are written according to your Print selection. The results can be printed directly to the printer or written to disk.</td>
</tr>
<tr>
<td>Calibration Run</td>
<td>Calibration Run lets you calibrate the model.</td>
</tr>
<tr>
<td></td>
<td>This option is used to determine the correct Average Path Length for batch workloads. More information on this option and calibration is contained in <a href="#">Chapter 6, “Getting started - performance modeling basics,” on page 27</a>.</td>
</tr>
<tr>
<td>Run Wizard</td>
<td>The Run Wizard provides a simple way to create and run multiple model configuration files (that is, SIM files).</td>
</tr>
<tr>
<td></td>
<td>This function should be used after you create a baseline model and now want to run multiple growth scenarios. You can also use this function to see the impact of upgrading to a new processor, or run combinations of growth and new processor scenarios.</td>
</tr>
<tr>
<td>Multi Image Run</td>
<td>This option should be used when you are trying to model multiple LPARs running on the same processor. Normally, you only model the performance of one LPAR. If there are two or more LPARs on the processor, the first LPAR (LPAR #1) is the only LPAR that is actually simulated. The other LPARs (LPAR #2 thru #15) are only...</td>
</tr>
</tbody>
</table>
Run options

represented as a fixed amount of mips consumed. This works fine when there is one dominant LPAR and multiple small additional LPARs. But when there are two or more dominant LPARs, you may want to model the performance of all of these LPARs. See "Multi Image Run" on page 17 for a description of this option.

The Run Wizard

The Run Wizard function is the primary way to describe and execute multiple modeling scenarios. It should be used to examine “what if” scenarios.

The first step to using the Run Wizard is the creation of a Baseline model. The Baseline model is used as the starting point for each new scenario. See Chapter 8, “Automatically generating a model with the Build option,” on page 45 for more information on how to automatically create a Baseline model.

When the Run Wizard is selected, Performance Modeler displays a Run Wizard data collection form (see Figure 4).

Figure 4. The Run Wizard screen

In Step 1 you specify the name of the Baseline Model. This is the file name containing the model definitions which are used as the starting point for generating all “what if” future scenarios.
The Run Wizard

In Step 2 you specify the Interval Type and Growth Rates for each interval. The Interval Type can be 3 months, 6 months, or Yearly. This field is for description only and specifies the time interval between each growth scenario. The # of Intervals (0-4) specifies the number of time intervals used to create each growth scenario. For example, if the Interval Type is Yearly, and the # of Intervals is 2, then a baseline model is created, followed by a Year 1 model, followed by a Year 2 model. That means there are a total of 3 models created for each processor selected.

The # of Growth Rates (0-5) specifies how many different Growth Rates (%) are applied for each time interval. You can display and change the Growth Rates by clicking the Display/Change Growth Rates command button. The specified Growth Rates are only used in models created for time intervals other than the Baseline run. The Baseline run represents the current time interval or interval # 0. If you specify the # of Time Intervals = 0, then the Growth Rates are ignored since only the Baseline runs are made. There are two Growth Rates you can specify. The first is the Growth Rate applied to the Model LPAR. This Growth Rate is applied evenly across all workloads in the Model LPAR (LPAR #1). The second Growth Rate is applied to the Other LPARs (LPARs #2-15). If you need to apply growth rates selectively to either the workloads or to each LPAR, then you should not use the Run Wizard.

In Step 3 you specify up to 8 processors to be modeled. The Baseline run, and each growth run is made for each processor selected in this step. You can add processors to the processor table by clicking the R (Replication) button and then manually changing the fields, or you can use the S (Selection) button to select a new processor. Once the Run command button is selected the Run Wizard begins building model definition files.

Before the models are run, the Wizard lets you alter the LPAR definitions for each processor you selected. This gives you the ability to change the number of Logical CPUs and Weighting Factors for each LPAR.

When you have entered all of the information, click Run. Performance Modeler runs each scenario, one behind the other, and writes the results to the Print option you selected.

At the completion of the runs, you have the option of deleting or keeping the Wizard generated model definition files.

Multi Image Run

The Multi Image Run option allows you to model the performance of up to ten LPARs all running on the same processor. When this option is selected you will be presented with the Multi Image Model Creation screen. [Figure 5 on page 18]
Before choosing this option you must create a baseline model for each LPAR you want to model. These LPARs can be currently running on the same or different processors. In Step 1 you enter the names of the configuration files for these input baseline models.

In Step 2 you choose the corresponding names for the models that will be generated. One new model will be generated for each input file name you specify.

In Step 3 you choose the processor that you want to model. Each LPAR will be modeled running on this processor along with all the other LPARs named in Step 1. In each new configuration file generated by this option, you will see multiple LPARs defined. The first LPAR represents the image to be modeled. LPARs #2 thru #10 represent the other LPARs running on this processor.

For example, if you want to model three LPARs running on a 2064-105, you would go through these steps:
1. Use the Build function to create three configuration files, one for each LPAR. These images may be currently running on the same or different processors.
2. Use the Multi Image Run option to specify each of these configuration files as input filenames. Choose appropriate filenames for the three output files that will be created. These files will be created as part of this option. Choose the 2064-105 as the processor to be modeled.
3. After filling in the fields and selecting Run, you will be presented with an LPAR Definition screen. This is your opportunity to change any of the LPAR definitions for each of these LPARs. The order of the LPARs on this screen will match the order you specified on the Multi Image Model Creation screen. That means the LPAR #2 definitions apply to the LPAR you specified as Image #2 on the previous screen. If you want to add additional LPARs (beyond the ones specified on the previous screen), you can add them here. These additional LPARs will not be modeled but their capacity requirements will be included in each model that is generated.

Once the LPAR definitions are filled in the simulator begins and a series of iterative runs are made. After each run, the capacity used by LPAR #1 will be inserted into the MIPs definition for that LPAR in the other models. That ensures that the correct MIPs values are used in each configuration file. This also ensures the performance of each LPAR is properly affected by the performance of the other LPARs. After all the models are run, the total utilization for each model is compared. When these are all within 1.25% of each other, the runs stop and the Multi Image Run option is completed. At this point you can run these models (using the Multiple Run option) to see the impact of running all of these LPARs on the same processor. If you want to make a change to a workload in any of these LPARs, or change the processor running these LPARs, you will have to rerun the Multi Image option and create all new configuration files. That is required because any change to a workload or processor can impact the capacity used by each LPAR.

The Simulator Running screen

The Simulator Running screen is displayed whenever the simulator is running. Figure 6 shows an example.
The Simulator Running screen

This information is shown on the Simulator Running screen.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>The Configuration field shows the file name of the model currently Running.</td>
</tr>
<tr>
<td>Simulation Time</td>
<td>Number of seconds that have been simulated to this point.</td>
</tr>
<tr>
<td>Real Time Used</td>
<td>Number of real seconds that have elapsed since the start of the run.</td>
</tr>
<tr>
<td>Physical CPU</td>
<td>Total CPU utilization (%) of the processor being modeled.</td>
</tr>
<tr>
<td>Model LCPU</td>
<td>Model Logical CPU % busy. When running in LPAR mode, this is the utilization of the model LPAR’s (LPAR #1) logical CPU configuration. This value can never be greater than 100 %. The calculation is CPU time used by Model LPAR / (Total Elapsed Time x # of Logical CPUs this LPAR)</td>
</tr>
<tr>
<td>Model PCPU</td>
<td>Model Physical CPU % busy. This is the percent of the total processor that is being used by the Model LPAR.</td>
</tr>
<tr>
<td>Workload</td>
<td>Workload Name.</td>
</tr>
<tr>
<td>X Count</td>
<td>Total # of transactions that have completed.</td>
</tr>
</tbody>
</table>
The Simulator Running screen

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU (%)</td>
<td>Single CPU % busy. Percent of a single CPU consumed by this Workload.</td>
</tr>
<tr>
<td>Page Flts</td>
<td>Page Faults. Total # of page faults that have occurred.</td>
</tr>
<tr>
<td>CPUR</td>
<td>Real CPU Time. This is the average CPU time (seconds) per Transaction.</td>
</tr>
<tr>
<td>NQ</td>
<td># of transactions currently on the input Queue. This is a snapshot view of how many transactions are waiting to run at this point in time.</td>
</tr>
<tr>
<td>AQLen</td>
<td>Average Queue Length. The average # of transactions on the input Queue measured over the total run time.</td>
</tr>
</tbody>
</table>
| Resp         | Response Time. This value has two meanings depending on the type of workload being modeled. For online workloads, this is the average response time in seconds. For batch workloads this value is called the Elongation Factor. Batch Elongation Factors represent a measure of average elapsed time for each batch job running in this workload. The Elongation Factor is measured by the simulator and is the # of real seconds that each batch job requires to execute 100 million instructions. Since the total number of instructions executed by a batch job is a constant value, the time required to run a batch job can be calculated by the following formula:

\[
\text{Elapsed Time} = \text{Elongation factor} \times \frac{\text{Total Number of Instructions}}{100 \text{ Million}}.
\]

In most cases, the Elongation Factor can be used by itself as a baseline for average batch performance. For example, if the first model shows an Elongation Factor of 25, and the second model shows an Elongation Factor of 50, the following conclusion can be drawn. The second model is predicting that average batch elapsed times double compared to the first model. For most modeling scenarios, the relative change in Elongation Factors is all that is needed to predict the relative change in batch performance.

Graphing workload performance

One of the options available while the simulator is running is the ability to see a graphical picture of workload performance. When the Graph 1 button is selected, you are presented with a panel where up to four workloads can be selected for graphing. Figure 7 is an example of the Graph Selection panel.
Graphing workload performance

Clicking on the box under Graph Select selects the corresponding workload for graphing. Clicking the box a second time cancels the selection.

When you select the OK button, Performance Modeler displays the Performance Tracking screen.

Each of the selected workloads is shown with its unique color. For online workloads, the average response time is graphed. For batch workloads, the graph shows the fraction of elapsed time that each batch job is executing instructions.

For example, if each batch job within the workload consumes 10% of a single CPU, the value being graphed is .10. This allows both online and batch performance to be shown on the same graph.

Figure 7 is an example of the Performance Tracking screen.
Workload performance is shown as a continuously updated line graph.

The + Scale and -Scale buttons allow you to double (+) or half (-) the Y axis scale. Changing the scale may be necessary to view performance metrics which are too large or small to fit on the default scale.

Selecting the Return button returns the view to the Simulator Running screen.

The Graph 2 button displays a graph showing the utilization of the processor, broken out by individual workload. Figure 9 shows an example of the graph displayed when you select the Graph 2 button.
Print options

When you select the Print button, Performance Modeler displays the Print Option screen. This screen lets you select where to direct the output of the simulation run. Output can be directed to a printer or to a disk file. Figure 10 shows an example of the Print Option screen.

Figure 9. The Processor Utilization Tracking screen
Two Disk File options are available, text and/or numeric.

If you choose the text file, Performance Modeler writes a text image of the output report to disk.

If you choose the numeric option, Performance Modeler generate a disk file where the results are separated by commas. This file can be imported into a spreadsheet as numeric data. Importing the simulation results into a spreadsheet makes it easy to do additional data manipulation and create customized graphs or tables of results.

The disk file name must be entered in the text box to the right of the option selected. If the text & numeric option is selected, you must enter file names for both types of output files.
Chapter 6. Getting started - performance modeling basics

Now that we’ve covered the basic operation and features of Performance Modeler, it’s time to discuss how to model real computer systems.

There are two ways to build a model:
1. You can manually enter all the input parameters using the three input panels.
2. You can use the Build option on the primary menu to automatically build the model.

This section describes the first, or manual build process. The automated Build option is described in Chapter 8, “Automatically generating a model with the Build option,” on page 45.

While reading through this chapter, you may feel that the process is overly complex. In fact, most models are built using the automatic Build procedure. This chapter is included to illustrate the tasks that must be done to accurately build a model. This gives you a better understanding of the functions provided by the Build option.

The manual build process has three steps, described below.

Step 1 Define the processor configuration

The processor configuration is defined using the Configuration Definition screen. The key input parameters are the CPU Speed (mips) and the # Of CPUs. The CPU Speed represents the capacity (MIPS) of a single CPU. Multiplying the CPU Speed by the # Of CPUs yields the total capacity of the processor being modeled.

MIPS Discussion

In order to properly interpret and use the MIPS values provided in Performance Modeler, consider the following:

Although capacity is often expressed as MIPS, no one, including IBM®, actually measures MIPS. All mainframe vendors measure relative capacity. IBM publishes these results in a table of relative capacity called the LSPR (Large Systems Performance Reference) results. These results show the relative capacity of many IBM and PCM processors based on a set of workload specific benchmarks.

But LSPR results are often converted into MIPS. That’s because MIPS have been used for years and are a recognized measure of processor capacity. Converting LSPR results into MIPS requires that you assign a MIPS rating to one processor, and extrapolate the MIPS ratings for the other processors using the LSPR ratios.

For example, assume the LSPR rating for processor B is twice the capacity of processor A. Then if we assign a MIPS rating of 100 MIPS to Processor A, we can extrapolate the MIPS rating of processor B to be 200 MIPS. This is how the CPU MIPS table available on the Configuration Definition screen was created. In fact, this table is based on assigning 63 MIPS to the IBM 9672-R15. All other MIPS ratings are created by extrapolation from the R15 using LSPR data.
Step 1 Define the processor configuration

From time to time, IBM has changed the actual workloads that make up the LSPR benchmarks. They do that to keep the benchmarks relevant and ensure they correctly match the kinds of work that customers actually run on their processors. Sometimes new results are produced using a newer version of the operating system, or reflect an architectural change, such as the change from 31 bit to 64 bit addressing. New LSPR results often change the capacity ratios for existing processor models. Plus, newer LSPR results may not include all of the processor models included in the earlier tables, especially older models that were not benchmarked with the latest workloads. This makes it difficult to produce a single table of MIPs for all processors. In Version 2 of Performance Modeler, the following changes (compared to Version 1) have been made to the MIPs table.

First, the new IBM processor models announced on May 13, 2003 have been added to the MIPs table. These include all the 2084 models, from the uniprocessor 2084-301 to the 32 way 2084-332. The MIPs assigned to these models was calculated using the LSPR Default Mixed Workload. This mix is based on a 20% (equal) mix of five LSPR benchmarks. These are the CB-L, CB-S, OLTP-T, OLTP-W, and WASDB workloads. If you need an explanation of these workloads as well as the detail capacity ratios for each workload, you can find this information at this IBM Web site

http://www.ibm.com/servers/eserver/zseries/lsp

Since these new results only included measurements for the z800, z900 and z990 models, in order to update the MIPs table it was necessary to "bridge" these new ratios back to the earlier ratios. That was done by choosing the 2064-1C1 as the bridge machine, and assigning it a MIPs rating of 252.8 MIPs (the same rating used in Version 1). Based on that rating, all of the other MIPs ratings were calculated using the LSPR ratios for the Default Mix workload. One result of this update is that many of the z800 and z900 MIP's ratings have changed slightly from Version 1.

One caution regarding use of these MIPs ratings. The MIPs table included with Performance Modeler is meant as a reference and starting point for determining relative capacity. A single table of relative capacity cannot show the variations based on different workload mixes or changes in operating systems. Plus, these ratings do not consider the impact of LPAR overhead. When modeling the impact of changing processor models, it is important that the relative capacity between the baseline model and the new processor model accurately reflects the change in capacity. You may choose to use the MIP's rating provided in the MIP's table for the baseline model, but you should be careful that the ratio of the new processor MIPs to the baseline processor MIPs be as accurate as possible. That is the only way to be sure you are properly modeling the change in processors. That may require changing the MIP's rating for the new processor from the value provided in the MIP's table.

One way to make sure you are using the correct ratio is to consult the actual LSPR ratings provided at the IBM Web site. If the LPAR configuration changes between the baseline and target models, you should consider adding a value to the LPAR Overhead field in the LPAR Definition screen. If you need help determining the impact of LPAR changes, contact your IBM representative and ask for their assistance.

MIPS may be an abstract term but it is very useful for modeling and capacity planning. For example, a single transaction is defined by the number of instructions (path length) it needs to execute. Once the path length is known, you can calculate the CPU time it takes to execute this transaction by this calculation:
Step 1 Define the processor configuration

CPU time = path length (in millions of instructions) divided by the single CPU MIPS rating

To further illustrate, consider a transaction with a path length of 1 million instructions. The CPU time it takes to execute this transaction on a CPU rated at 50 MIPS is $1/50 = .02$ seconds. This calculation is used by the simulator, so the MIPS rating and the average transaction path length are linked by this relationship. You can select the MIPS rating by using the pull down CPU table in the Configuration Definition screen. Or you can choose your own rating.

Step 2 Selecting workloads to be modeled

Choosing the workloads to be modeled requires that you understand how your processor is being used. You do not need to model every workload that runs on your processor. In fact, you can only define up to 20 workloads in one model. The recommended strategy is to review workload and CPU utilization reports.

If you have RMF (Resource Measurement Facility) installed, you can run the CPU and Workload Activity Reports for the peak times of processor utilization. The CPU reports show processor utilization.

If you run multiple LPARs, the CPU report shows processor utilization for each LPAR. The Workload Activity Report shows processor use by workload. Workloads are reported differently depending on whether the system is running in Compatibility Mode or in Goal Mode.

When systems run in Compatibility Mode, workloads are classified and reported by their Performance Group Number. If the system runs in Goal Mode, workloads are reported by Workload Name and by Service Class Name. Workloads may also be defined and tracked by Reporting Group Number (Compatibility Mode) or by Reporting Class (Goal Mode). In either case, these are useful ways to classify the different workloads when building a model.

By scanning these reports you can see which workloads use the largest amount of the processor. These big users are prime candidates when choosing the workloads to be modeled. You may also want to choose the most important workloads even though they may not be the biggest users of the processor.

Figure[11] shows an example of a Workload Activity Report for one Performance Group Number.
Step 2 Selecting workloads to be modeled

Figure 11. A workload activity report

In this example, data is shown for Period 4 (PGP) of Performance Group Number 2 (PGN). The percent of the time that this workload was executing instructions is 24.5% (from the APPL% field). This number can be converted into percent of the total processor used by this workload by this calculation:

- This processor is a 9021-982 which has 8 physical CPUs.
- Since all 8 CPUs are available for execution in each time interval, the percent of total processor used by this workload is 24.5%/8 = 3.06%.
- Multiplying CPU % busy by the MIPS rating for this processor gives you the MIPS consumed by this workload.
- If the 982 is rated at 408 MIPS, then this workload must be consuming .0306 x 408 = 124.8 MIPS.

After scanning all the workloads in the report, you can create a table that shows how capacity is consumed. You are now ready to select the workloads you want to model.

Step 3 Define the Fill in workloads

After you select the workloads you want to model, you need to define the workloads that fill in the gaps. These are workloads that represent all the other users of capacity.

For example, assume you want to model two workloads. The first workload runs high priority and the second runs lowest priority. To model this correctly you need to specify at least three workloads. The first two workloads are the ones you are interested in, but you must also account for all the other work and the MIPS they consume. The MIPS consumed by these other workloads can be concentrated in a single composite workload. This is the technique that allows the model to simulate complex systems while being limited to a maximum of twenty workloads.

Before you can define workload parameters, you must prioritize all workloads by dispatching priority. For systems that run in Compatibility Mode, the dispatching priority is assigned to each Performance Group in the IEAIFSxx list in the SYS1.PARMLIB data set. If the system runs in GOAL Mode, the dispatching priority is dynamically adjusted. But each Service Class is assigned an Importance relative to other Service Classes, and that can be substituted for dispatching priority.

In Figure[12] a table of workloads has been created and sorted by dispatching priority. As you can see, the workloads are defined by Performance Group Number 30
**Step 3 Define the Fill in workloads**

(PGN) so this system is running in Compatibility Mode. This table serves as the basis for selecting the workloads to be modeled. This table can be built by manually inspecting the RMF Workload Activity Reports, or it can be generated automatically (the preferred way) by the Build option (see Chapter 8."

"Automatically generating a model with the Build option,” on page 45).

Workload Analysis for CPU1 (PRODMVS1) on 7/06/98 at 11:30 thru 11:45 AM

<table>
<thead>
<tr>
<th>Workload</th>
<th>PGN</th>
<th>Period</th>
<th>Priority</th>
<th>CPU (%)</th>
<th>Total (%) Uncptrd=</th>
<th>Cumulative</th>
<th>MPL</th>
<th>Xact/Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest Priority</td>
<td>0</td>
<td>1</td>
<td>255</td>
<td>9.70</td>
<td>1.08</td>
<td>10.78</td>
<td>9.00</td>
<td>0.00</td>
</tr>
<tr>
<td>JES2/OMEGAMON</td>
<td>9</td>
<td>1</td>
<td>139</td>
<td>42.50</td>
<td>4.72</td>
<td>15.50</td>
<td>6.00</td>
<td>0.00</td>
</tr>
<tr>
<td>VTAM</td>
<td>5</td>
<td>1</td>
<td>135</td>
<td>7.00</td>
<td>0.78</td>
<td>16.28</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>STARTED TASKS</td>
<td>11</td>
<td>1</td>
<td>135</td>
<td>122.50</td>
<td>13.61</td>
<td>29.89</td>
<td>49.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPF</td>
<td>36</td>
<td>1</td>
<td>122</td>
<td>4.50</td>
<td>0.50</td>
<td>30.39</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPH</td>
<td>38</td>
<td>1</td>
<td>122</td>
<td>33.90</td>
<td>3.77</td>
<td>34.15</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPI</td>
<td>39</td>
<td>1</td>
<td>122</td>
<td>73.20</td>
<td>8.13</td>
<td>42.29</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ1</td>
<td>40</td>
<td>1</td>
<td>122</td>
<td>6.00</td>
<td>0.67</td>
<td>42.95</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ2</td>
<td>41</td>
<td>1</td>
<td>122</td>
<td>11.40</td>
<td>1.27</td>
<td>44.22</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ3</td>
<td>42</td>
<td>1</td>
<td>122</td>
<td>23.40</td>
<td>2.60</td>
<td>46.82</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ4</td>
<td>43</td>
<td>1</td>
<td>122</td>
<td>23.70</td>
<td>2.63</td>
<td>49.45</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ5</td>
<td>44</td>
<td>1</td>
<td>122</td>
<td>17.00</td>
<td>1.89</td>
<td>51.34</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ6</td>
<td>45</td>
<td>1</td>
<td>122</td>
<td>21.50</td>
<td>2.39</td>
<td>53.73</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ7</td>
<td>46</td>
<td>1</td>
<td>122</td>
<td>4.90</td>
<td>0.54</td>
<td>54.28</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ8</td>
<td>47</td>
<td>1</td>
<td>122</td>
<td>28.00</td>
<td>3.11</td>
<td>57.39</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ9</td>
<td>48</td>
<td>1</td>
<td>122</td>
<td>21.90</td>
<td>2.43</td>
<td>59.82</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ10</td>
<td>49</td>
<td>1</td>
<td>122</td>
<td>24.10</td>
<td>2.68</td>
<td>62.50</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ11</td>
<td>70</td>
<td>1</td>
<td>122</td>
<td>22.70</td>
<td>2.52</td>
<td>65.02</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ12</td>
<td>71</td>
<td>1</td>
<td>122</td>
<td>26.70</td>
<td>2.97</td>
<td>67.99</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ13</td>
<td>72</td>
<td>1</td>
<td>122</td>
<td>27.50</td>
<td>3.06</td>
<td>71.04</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPZ14</td>
<td>73</td>
<td>1</td>
<td>122</td>
<td>30.20</td>
<td>3.36</td>
<td>74.40</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPK</td>
<td>74</td>
<td>1</td>
<td>122</td>
<td>0.60</td>
<td>0.07</td>
<td>74.47</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PRODPV</td>
<td>85</td>
<td>1</td>
<td>122</td>
<td>63.20</td>
<td>7.02</td>
<td>81.49</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>TSO 1ST PERIOD</td>
<td>8</td>
<td>1</td>
<td>93</td>
<td>1.70</td>
<td>0.19</td>
<td>81.68</td>
<td>0.19</td>
<td>0.00</td>
</tr>
<tr>
<td>TSO 2ND PERIOD</td>
<td>8</td>
<td>2</td>
<td>91</td>
<td>1.20</td>
<td>0.13</td>
<td>81.81</td>
<td>0.26</td>
<td>0.36</td>
</tr>
<tr>
<td>TSO 3RD PERIOD</td>
<td>8</td>
<td>3</td>
<td>90</td>
<td>2.70</td>
<td>0.30</td>
<td>82.11</td>
<td>0.43</td>
<td>0.44</td>
</tr>
<tr>
<td>TEST CICS/IDMS</td>
<td>4</td>
<td>1</td>
<td>63</td>
<td>2.00</td>
<td>0.22</td>
<td>82.33</td>
<td>0.92</td>
<td>0.00</td>
</tr>
<tr>
<td>TEST BATCH 1ST P</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>0.80</td>
<td>0.09</td>
<td>82.42</td>
<td>0.27</td>
<td>0.00</td>
</tr>
<tr>
<td>TEST BATCH 2ND P</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>87.20</td>
<td>9.69</td>
<td>92.11</td>
<td>6.99</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Total 82.41
Actual 92.11
Uncaptured 9.70
C. R. 89.47%
All 99.22%

Figure 12. Workload analysis

The data shown in this table includes:

**Workload** The name of the workload. This can be found in the comment section of the IEAIPSxx member of SYS1.PARMLIB. The Workload name is usually assigned on the same line that defines the dispatching priority to this Performance Group Number.

**PGN** Performance Group Number

**Period** Period for this Performance Group Number.

**Priority** Dispatching Priority as a decimal number. The dispatching priority is assigned in the IEAIPSxx member of SYS1.PARMLIB. The higher the number, the higher the priority.

**CPU (%)** Percent of a single CPU consumed by this workload.

**Total (%)** Percent of the total processor consumed by this workload.

**Cumulative** Percent of the processor consumed by this workload plus all workloads that run higher priority.
Step 3 Define the Fill in workloads

**MPL**
Multi-Programming Level. This is the average number of jobs running in the workload during the reporting interval.

**Xact/Sec**
Transactions per second. This is the average transaction rate for this workload. This value is zero for long running tasks that do not start or stop during this interval.

At the bottom of the table the Total (%) column is summed. This is the total CPU % busy captured for each reported workload. The Actual value is the actual CPU % busy reported for the entire system or LPAR. This information is reported in the RMF CPU report. The difference between Total and Actual is the Uncaptured time. C. R. is the capture ratio, the percent of actual utilization that is captured and assigned to a workload (Total/Actual). And finally, the All field contains the total utilization for the entire processor. This table contains the information that is needed to build a model. All the information shown in this table is available in either the RMF Workload Activity Report, the RMF CPU report, or in the IEAIPSxx member in SYS1.PARMLIB.

We are now ready to choose the model input parameters for the workloads being modeled. In the workload table you can see some of the workloads are printed in bold type. These have been selected because they represent the largest consumers of CPU time. Figure 13 is another table which was created from the workload table above. This table is used to calculate the actual modeling input parameters for each workload.

**Simulator Input (based on 7/06/98 at 11:30-11:45 AM)**

<table>
<thead>
<tr>
<th>Workload</th>
<th>Priority</th>
<th>Single CPU</th>
<th>Total CPU</th>
<th>MIPS</th>
<th>Path</th>
<th>Tran/Sec</th>
<th>I/Os / Tran</th>
<th>Type</th>
<th>MPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hi Prty</td>
<td>1</td>
<td>269.01%</td>
<td>29.89%</td>
<td>96.25</td>
<td>1.00</td>
<td>96.25</td>
<td>50.00</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>PRODP1</td>
<td>2</td>
<td>73.17%</td>
<td>8.13%</td>
<td>26.18</td>
<td>1.00</td>
<td>26.18</td>
<td>50.00</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>PRODPV</td>
<td>2</td>
<td>63.18%</td>
<td>7.02%</td>
<td>22.60</td>
<td>1.00</td>
<td>22.60</td>
<td>50.00</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>PRODPI4</td>
<td>2</td>
<td>30.24%</td>
<td>3.36%</td>
<td>10.82</td>
<td>1.00</td>
<td>10.82</td>
<td>50.00</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>OTHERPROD</td>
<td>2</td>
<td>297.81%</td>
<td>33.09%</td>
<td>106.55</td>
<td>1.00</td>
<td>106.55</td>
<td>50.00</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Fill</td>
<td>3</td>
<td>8.37%</td>
<td>0.93%</td>
<td>2.99</td>
<td>1.00</td>
<td>2.99</td>
<td>50.00</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Test Batch</td>
<td>4</td>
<td>87.21%</td>
<td>9.69%</td>
<td>31.20</td>
<td>0.20</td>
<td></td>
<td></td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>92.11</td>
<td>296.59</td>
</tr>
<tr>
<td>Other LPARs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.11</td>
<td>22.09</td>
</tr>
<tr>
<td>All</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>99.22</td>
<td>319.49</td>
</tr>
</tbody>
</table>

Assumptions: 9672-004 = 322 MIPS
1 Million Instructions per Transaction for Online Workloads
50 Disk I/Os per 1 Million Instructions for Online Workloads

**Figure 13. Simulator input**

This table shows the workloads we want to model, but also shows a workload called Hi Prty and a workload called Fill. The Hi Prty workload is a composite workload which includes all the workloads that run higher priority than PRODP1. The Fill workload is also a composite and represents all the workloads that run lower priority than OTHERPROD and higher priority than Test Batch. OTHERPROD is also a composite and represents all of the workloads that run equal priority with the PROD workloads being modeled.

The columns include:

**Workload** Workload name

**Priority** In this column we have assigned a priority such that 1 is highest
Step 3 Define the Fill in workloads

and 99 is lowest. This is the reverse from the previous table but this is the format used by the model (lower the number, higher its priority).

**Single CPU**  
Single CPU % busy. This is equivalent to the column labeled CPU (%) in the previous table.

**Total CPU**  
Total processor % busy. From the column labeled Total (%) in the previous table.

**MIPS**  
The total MIPS consumed by this workload. This is the product of the Total CPU and the MIPS rating assigned to this processor. At the bottom of the table you see a comment that identifies the processor as a 9672-R94 and assigns a rating of 322 MIPS to this processor.

**Path**  
This is the average path length (instructions per transaction) that is assigned to transactions that are generated for this workload. In most cases, the average transaction rate is not known. In that case it is appropriate to estimate a path length based on experience or other rules of thumb. Most online workloads have transaction path lengths that vary from 300,000 instructions to over 2 million instructions.

Choosing 1 million instructions has some obvious advantages. First it is a reasonable estimate for most complex transactions. And second, it simplifies the calculation of transaction rate. When average path length (x 1 million) is multiplied by average transaction rate (trans per second), the product is MIPS consumed. Thus when you assume a path length of 1 million instructions, you can calculate the transaction rate as being the MIPS consumed.

**Trans/Sec**  
Average transaction rate. This is calculated as MIPS divided by path Length.

**I/Os / Tran**  
Number of I/O operations per transaction. This parameter may also be difficult to obtain. In this example, 50 I/Os are assumed per every 1 million instructions. This is another rule of thumb and can be used when the actual value is not known.

**Type**  
Type of workload. This is either M for Multi-tasking workloads, S for Single tasking workloads, and B for batch workloads. In this example, all the composite workloads are defined as type M. This is because this workload mimics a number of real workloads where transactions can run across multiple CPUs at the same time. The PROD workloads represent individual CICS regions and are treated as type S workloads. The lowest priority workload labeled Test Batch represents batch jobs and are defined as type B.

**MPL**  
Multi-programming level. This model parameter is usually only specified for batch workloads. As described earlier, this represents the average number of batch jobs to be modeled for a given workload. This can be a non-integer value such as 4.5 and is derived from the MPL field in the workload activity report.

The data shown in the table created the model shown in these three screens of model input parameters:
Step 3 Define the Fill in workloads

![Model input - the Configuration Definition screen](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE FOR THIS RUN:</td>
<td>Model for 9021-982 CPU1 (PRODMVS1)</td>
</tr>
<tr>
<td>TOTAL RUN TIME (sec):</td>
<td>200</td>
</tr>
<tr>
<td>TIME INTERVAL (sec):</td>
<td>0.0100</td>
</tr>
<tr>
<td># OF SECONDS PER REPORT:</td>
<td>6</td>
</tr>
<tr>
<td>CPU DESCRIPTION:</td>
<td>9021-982</td>
</tr>
<tr>
<td>CPU SPEED (mips):</td>
<td>61.06</td>
</tr>
<tr>
<td># OF CPUs (1-32):</td>
<td>8</td>
</tr>
<tr>
<td>PAGE PACK RESP. TIME (sec):</td>
<td>0.020</td>
</tr>
<tr>
<td>E-STOR. RESP. TIME (sec):</td>
<td>0.000030</td>
</tr>
<tr>
<td># OF WORKLOADS (1-20):</td>
<td>9</td>
</tr>
<tr>
<td># OF LPARS (0-15):</td>
<td>2</td>
</tr>
</tbody>
</table>
**Step 3 Define the Fill in workloads**

![Workloads Definition screen](image)

**Figure 15. Model input - the Workloads Definition screen**
Handling high transaction rates

The input parameters shown in these screens were derived from the data in the Simulator Input table. Notice that there are two workloads defined as Hi Prty in the Workload Definition screen. There are also two workloads defined as OTHRPROD. These workloads are duplicated and spread over two definitions because of a limitation in how the simulator works. As described in the Simulator Overview, simulation requires that real time is divided into discrete increments of time. The size of this time increment is specified in the Time Interval field in the Configuration Definition screen. In the example above, this interval is .01 seconds. That means a real second is divided into 100 increments. But it also means the maximum number of times a workload can generate a new transaction is 100 times each second.

This limits the maximum generated transaction rate to 100 transactions per second. In this modeling example, the Hi Prty workload must generate 96.25 transactions per second. While 96.25 is less than 100, it is close enough that the model tries to generate a new transaction nearly every pass of the simulator or every .01 seconds. The result is that transactions do not appear to be arriving in a random pattern and could bias the modeling results. The solution is to split the workload into two similar workloads, each running half the target transaction rate, and each running at the same dispatching priority. The OTHRPROD workload also has a large target transaction rate (106.55). And like Hi Prty, this workload is spread over two identical workloads, each running half the transaction rate.
Handling high transaction rates

A good rule of thumb is to split workloads into multiple equal workloads whenever the target transaction rate is 85% or greater than the number of time intervals per second. Another solution to this problem would be to reduce the Time Interval. For example, changing the Time Interval from .01 to .001 seconds would mean the simulator makes 1,000 iterations per second of simulated time. This also means each workload could generate up to 1,000 transactions per second. Although this change would increase the transaction rate it would also increase the elapsed time it takes to run the model. In most cases, the recommended solution is to leave the Time Interval at .01 seconds, and make a copy of the workload with the high transaction rate. When you start the simulator, the program checks to see if any workloads are defined with Arrival Rates that are greater than 85% of the maximum that can be generated. If found, you are warned, and allowed to cancel the simulation run, or to continue it.

Batch workload calibration

You may have noticed that the average path length for the batch workload is shown as .200 or 200,000 instructions, not the 1 million instructions which is used for online workloads. This is because batch workload path lengths have a different meaning compared to online workloads. For batch workloads, path length represents the average number of instructions executed before stopping to perform an I/O operation. This is also a rule of thumb value, but it is really used as a starting Point.

All batch jobs are modeled as a never ending transaction. These transactions execute a number of instructions (path length), then stop to perform an I/O operation, then resume executing. The MIPS consumed by batch jobs depend on several factors including the ratio of path length to I/O delay. This ratio determines whether the job is CPU or I/O bound. The greater the ratio of path length to I/O delay, the more CPU bound and the more MIPS are consumed by the job. But the RMF Workload Report only shows the CPU (%) consumed, and the MPL (multi-programming level) for each batch workload. While the actual average path length for these jobs may not be known, you can use the model to back into the path length by executing the Calibration Run option (explained in the following section).

Select Calibration Run from the primary menu Run option. The next screen shown is the Calibration Form screen. Figure 17 is an example of a Calibration Form screen.
Batch workload calibration

This form lets you set the target single CPU percent busy for each batch workload defined to the model. As you can see, input fields are only shown for batch workloads. In the figure, the only batch workload is TstBtch.

The single CPU % busy target should be changed from the default of 20% to the value shown in the Simulator Input table, 87.21%. The next field, Max % Difference, is the amount of error tolerated in calibrating this workload.

During the calibration run, the actual measured single CPU % busy is compared with the target. If the difference between actual and target exceeds the Max % Difference, the path length is adjusted and the calibration is restarted from the beginning. For example, if the measured % busy is higher than the target, the path length is lowered. This has the effect of making the batch job less CPU bound and more I/O bound.

If the measured % busy is too low, the path length is increased. The calibration process continues until all batch workloads are within the Max % Difference set by you. For most models, the default Max % Difference of 2% is sufficient.

The Number of Seconds per Calibration is the number of simulated seconds that must be modeled before the measured % busy is compared to the target % busy. In this example, 40 seconds is chosen as the interval between calibrations. Once again, the default of 40 seconds should provide good results for most models.
As a result of these settings:

- When you click the OK button, the model begins the simulation process.
- After 40 seconds of simulated time has passed, the measured single CPU % busy for each batch job is compared to its target % busy.
  - If the difference between measured and target % busy are within the Max % Difference value for all workloads, the calibration ends.
  - If at least one workload is not within the Max % Difference, all path lengths are adjusted and the model begins running again from the beginning.
- While running, the Max % Difference for the workload which is furthest away from its target is displayed on the top of the Simulation Running screen. This provides a simple way to track the progress of the calibration run.
- At any time you can terminate the calibration by clicking the Stop button.

After the calibration completes, you can see the calibrated path lengths for each batch workload by viewing the Workload Definition screen. These are the path lengths which enable these batch workloads to consume the correct amount of CPU time. You can now save this model to disk using the File Save option.
Chapter 7. Extracting RMF report data

The Extract option provides a data reduction program which reads RMF (or CMF) reports and extracts important performance metrics. The extracted files serve two functions. First, they can be imported into spreadsheets for further data analysis. Spreadsheets are excellent tools for analyzing performance metrics and creating presentation tables and graphs based on this data. More information on how Performance Modeler works in synergy with spreadsheets is provided in Chapter 10, “Synergy with spreadsheets,” on page 61. The extracted files are also used by the Build option to automatically build a model. This function is described in Chapter 8, “Automatically generating a model with the Build option,” on page 45.

The extracted files are created in a format that can be easily imported into a spreadsheet as numeric data. For example, the extract files contain fields which are separated by commas. If the field is a literal, it is enclosed by double quotation marks. When the file is imported into a spreadsheet, each field is stored in its own cell. Literals are stored as text while numeric data is stored as numeric. Before running the Extract option, the RMF reports must be downloaded from the mainframe to the PC in text format.

When the Extract option is chosen, the RMF Extraction screen is displayed.
The Report Type refers to the type of extract file which is created. The LPAR Report creates an extract of CPU utilization and shows the breakout of utilization by LPAR. The input file is specified in the RMF Input File Name field. This must be the name of an RMF (or CMF) Postprocessor report which contains a CPU Report. The RMF Report file must be in text (not Binary) format. The LPAR Report File Name is the name assigned to the extracted output file. By convention, extract reports are given the file type RPT. That makes it easy to locate these files when they are required during the Build option.

The LPAR Extract report can include information for all of the LPARs with shared CPUs, or it can be for a single LPAR with Dedicated CPUs. If the Extract program detects there are LPARs with Dedicated CPUs it displays a list of the Active LPARs. You are prompted to choose which LPAR is input to the Extraction program. Choosing any LPAR with Shared CPUs results in the Extraction of data for all the Shared LPARs. Choosing a Dedicated LPAR results in Extraction for that LPAR alone.

The Compatibility Workload Report option extracts data from an RMF (or CMF) Workload Activity Report. Workload Activity Reports are created for system images running in compatibility mode. When this option is selected, you are also prompted to provide the names of the files containing the image’s IPS and ICS files (from SYS1.PARMLIB). The IPS and ICS files are used together with the RMF Workload Activity Reports to generate the extracted Workload Reports. These reports contain a breakout of CPU used by each Performance Group and Period.
within Performance Group for each time interval captured in the RMF report. The RMF input file must contain a Workload Activity Report at the detail level (Period within Performance Group Number). Here is an example of RMF post processor control cards for producing the CPU and Workload Reports in a system running in compatibility mode:

SYSID(XXXX)
SYSOUT(M)
DATE(01012000,12312000)
RTOD(1000,1100)
STD0(1000,1100)
REPORTS(CPU)
REPORTS(WKLD(PERIOD))

The Goal Mode Workload Report option is used when the system image runs in goal mode. This provides an extract of CPU utilization by Service Class and Period within Service Class. The RMF report that is input to this option must include a Goal Mode Workload Activity Report at the Period within Service Class. Use the SCPE Parm in the RMF Report selection control statement. If multiple images run on the processor and they are part of a SYSPLEX, the RMF report combines these images into a single report. You can instruct RMF to produce the workload report for a specific system image by using the SYSNAM parameter. Here is an example of RMF post processor control cards that produces the workload report in the correct format:

SYSID(XXXX)
SYSOUT(M)
DATE(01012000,12312000)
RTOD(1000,1100)
STD0(1000,1100)
REPORTS(CPU)
SYSRPTS(WLML(SCPER, RCLASS, SYSNAM(XXXX)))

The SCPE Parm requests a Workload Report at the Period within Service Class level. The RCLASS Parm requests that a report by Reporting Class is also created. The SYSID(XXXX) and SYSNAM(XXXX) parms request that the report be only for the image with SMFID=XXXX. If the system is part of a Parallel/Sysplex, the SYSNAM Parm must be used. Using SYSNAM ensures that the CPU time reported for each Service Class is only reported for the system image being examined.

Clicking the S command button besides each input field displays a list of file names in the current directory. Selecting (single left click) a file name in the display list inserts that file name into the input field. This ensures that the file exists (for input files) and is spelled correctly. The Filter on SYSID option can be used when the RMF report contains several reports for different system images (SYSIDs). By selecting Yes, you are asked to provide a four character SYSID. Only reports for the selected SYSID are extracted.

Extract error messages

During the Extract Process you may see error messages that indicate a problem was detected. These messages typically begin with “Couldn’t find ...”. This means the input RMF or CMF report did not contain certain literals or data fields that the extract program expected to find. If you encounter one of these errors, take a look at the input report and make sure it was created correctly. If you cannot fix the problem, write down the error message and report the problem to IBM.
Chapter 8. Automatically generating a model with the Build option

Building a baseline model is the most critical part of any capacity planning effort. All “what if” scenarios and future modeling runs are compared back to the results in the baseline. Once the baseline is built and calibrated, it is relatively simple to change configuration definitions such as processor model or LPAR parameters. This is why so much space was given in Chapter 6, “Getting started - performance modeling basics,” on page 27 to explain how input parameters are chosen. The manual model build process described in Chapter 6 requires several steps and a great deal of manual analysis. The automated Build process described below simplifies this process and automates several of these steps.

CPU and Workload reports

The automated Build process begins when you select the Build option on the primary menu. The next form shown is the Model Build Form.

![Model Build Form](image)

This form shows five steps that are required to complete the Build process. These are described below:
CPU and Workload reports

Step  Description

Step 1 — Select processor
This step asks you to define the CPU being modeled. The key parameters are the MIPS rating and the number of CPUs.

Step 2 — Select CPU report file
In Step 2, you specify the name of the CPU report file. This is the same file created during the LPAR Report Extract function described in Chapter 7, “Extracting RMF report data,” on page 41. This file contains a table of CPU utilization for each LPAR, for each time interval captured in the RMF CPU report. Single clicking the Step 2 Command Button displays a list of valid file names (*.RPT). Single clicking a file name in the display list inserts that file name into the input field.

Step 3 — Select workload report file
In Step 3, you specify the name of the extracted workload reports created by the Workload Report Extract function. This report contains CPU utilization by workload for each time interval.

Step 4 — Select date and time
Step 4 displays the Date/Time Selection screen and shows CPU utilization, by LPAR, for each reported time interval. You are asked to select the LPAR and the time interval which you want to model. This is done by clicking on the cell containing the LPAR CPU utilization.

Step 5 — Select workloads for modeling
Step 5 shows the Workload Selection screen for the selected LPAR and time interval. This report is identical to the Workload report shown in Chapter 6. You are now prompted to select the workloads that you want to model. This is done by clicking anywhere on the row for each workload. When complete, click the Build button. The model input parameters are automatically generated.

Figure 20 shows an example of the CPU report displayed during Step 4.
In this example, the CPU Report shows there are two LPARs. The column labeled PROD contains the CPU % busy for one of the two production LPARs, where each row represents a different reporting interval. Clicking a cell under PROD selects that LPAR and the corresponding date and time for modeling.

Figure 21 is an example of the Workload Selection screen shown during Step 5.
Clicking anywhere on a row on this form selects that workload for modeling. When you click on a row, another form is displayed and ask for input on the workload Type. You can select M (Multi-tasking), S (Single tasking), or B (Batch). You can also change the name of the workload. This is the name used in the Workload Definition screen. When a workload is selected, an X appears in the first column for that workload, followed by an M, S, or B to indicate workload type. Clicking on the same workload a second time cancels the selection for that workload.

The automated Build process builds workload definitions for all the selected workloads as well as the Fill workloads. Fill workloads are needed to account for all the other workloads that run with the selected workloads. These include higher as well as equal and lower priority workloads. The Build process also generates a second LPAR whenever it detects LPAR mode. This LPAR is defined as containing all the work that runs in all the LPARS except for the LPAR being modeled (LPAR #1). Although there may be more than two LPARs running on the real processor, the Build process lumps all the LPARs (other than LPAR #1) into a single composite LPAR (LPAR #2). The MIPS field for this LPAR is equal to the difference between total MIPS consumed and the MIPS consumed by LPAR #1.

Following a successful Build operation, you are prompted to change the LPAR definition parameters. The LPAR definition screen is initially set with default values. This is your chance to fill in the correct values for Number of Logical CPUs and Weighting Factors for each LPAR. You should adjust these values to match the
actual LPAR definitions used on the system being modeled. Only two LPARs are
defined (when LPAR mode is active). These are LPAR #1 (the model itself), and
LPAR #2. LPAR #2 is a composite LPAR which includes all the LPARs other than
the model LPAR.

For LPAR #1 the #LCPUs field should be adjusted to be the same as the number of
Logical Processors as defined for the partition in the Partition Data section of the
RMF CPU Activity Report. For LPAR #2 the field should be adjusted to equal the
total number of logical processors for all other partitions than the one being
modeled providing that does not exceed the number of shareable physical CPUs. If
the total exceeds the number of shareable CPUs then this field should be set to the
maximum number of shareable CPUs. The number of sharable physical CPUs is
the number of installed physical CPUs minus any CPUs assigned as dedicated
processors to a partition.

The Weighting Factor assigned to LPAR #1 should be the same as the weighting
factor as defined in the Partition Data section of the RMF CPU Activity Report.
The Weighting Factor assigned to LPAR #2 should be changed to the sum of the
Weights for all of the LPARs other than LPAR #1.

The MIPS assigned to LPAR #2 is the total MIPS consumed by all of the LPARs
other than LPAR #1.

After LPAR definitions are defined and if Batch workloads have been selected, the
model must be calibrated to ensure that Batch path lengths are set correctly. After
the LPAR Definition screen is completed, the Build process displays the Calibration
screen. You can change the defaults set in the Calibration screen or use the ones
shown. The Single CPU % Target in the Calibration form is set to the actual CPU
% used by that workload during the chosen interval. Following the Calibration
Run, the Build process is complete and the modeling parameters can be saved to
disk. This model represents the baseline for the chosen report interval and can
serve as the starting point for modeling future “what if” scenarios.

After the LPAR Definition screen is completed and if no Batch workloads have
been selected, the Build process displays the Configuration screen.

---

**Defaults used by Build**

The Build process uses several defaults to build the model. These defaults should
be reviewed to see if they should be changed. For example, the I/O response times
are based on the average disk response times reported in the RMF Workload
Activity reports. Occasionally, these values may be skewed by a single disk drive
and may not make sense for inclusion in the model. An example would be a disk
response time which is exceptionally high such as greater than 1 second. When this
occurs, it probably makes sense to manually change it to a more realistic value. If
the average disk response time cannot be found in the workload reports, the Build
process uses a default of .010 seconds.

If the SSCHRT (Start Sub Channel Rate) is available, this will be used to calculate
the I/Os per online transaction. If the SSCHRT is not available, Build will use a
default of 50 disk I/Os per online transaction. If you know the actual number of
I/Os per transaction, you should change it to match the real value. Paging rates,
for auxiliary and Expanded storage paging defaults to 0. If you are experiencing
significant paging rates, change these values to match the actual paging rates.
Build also uses a default of one million instructions per online transaction. Once the path length is set, Build calculates the average transaction rate by using the following relationship. The MIPS consumed by this workload is the product of average path length and the average transaction rate. Once the MIPS consumed value is known, and the average path length is set to the default value of one million instructions per transaction, the transaction rate is simply calculated as MIPS consumed divided by one. For example, if a workload consumes 45 MIPS and the average path length is set to one million instructions per transaction, the transaction rate must be 45 transactions per second.

If you measure the actual CPU time used by an average transaction you may elect to change this default. For example, assume the Build process generated these model parms:

- An average path length of one million instructions per transaction
- An average transaction rate of 50 transactions per second

This means the workload consumes 50 MIPS. Now assume that actual measurements show the average CPU time per transaction is .02 seconds, and assume each CPU is rated at 120 MIPS per CP. Then multiplying 120 MIPS by .02 seconds per transaction equals the actual path length of 2.4 million instructions per transaction. If you change the default path length from 1.0 to 2.4 you must also change the transaction rate. In order to keep the MIPS consumed the same, you must change the average transaction rate by dividing the original value by 2.4. That means the transaction rate must be changed from 50 to 50/2.4 = 20.83 transactions per second. The new path length multiplied by the new transaction rate = 2.4 x 20.83 = 50 MIPS, so the MIPS consumed does not change.
Chapter 9. Capacity planning with Performance Modeler

Capacity Planning is one of several Systems Management disciplines. It includes the collection of activities that enable IT shops to deliver acceptable service while also managing their IT costs. Despite numerous books and papers on the subject, Capacity Planning today is often given the least amount of attention compared to other Systems Management functions. The reasons for this include 1-lack of trained staff, and 2-lack of effective tools. The result of poor Capacity Planning can include poor service delivery and the resulting dissatisfaction among the user community. It can also result in sudden unplanned upgrades with the usual pressure on the IT budget. These results create a climate where the IT organization appears to be in react mode and unable to plan proactively for future upgrades.

Capacity Planning has many definitions. An appropriate one is “The activities that ensure there is enough capacity to satisfy your end user’ service requirements.” For online workloads, service is often measured as average response times. For batch workloads, service can be defined as elapsed times for individual or groups of related jobs. It may also be appropriate to measure service as the time it takes to complete a nightly batch cycle. For example, many shops must complete a series of batch jobs during third shift before online workloads are started in the morning. But whatever the measure of service, each shop can describe a set of service level objectives that define the line between acceptable and unacceptable service.

Service Level Agreements (SLAs)

IT shops that do a good job of Capacity Planning formalize their service level requirements into a set of Service Level Agreements. SLAs are contracts where the end user agrees with IT management on what are acceptable levels of service, as well as the remedy if the service level is not met. But most important, SLAs can form the basis of a Capacity Planning methodology. Simply put, capacity should be managed such that Service Level Agreements are always being met. But this methodology requires that Capacity Planners be able to predict how service delivery changes over time. This includes predicting the impact of adding new workloads, or growth to existing applications, or the impact of configuration changes. These requirements point out the need for tools which can predict how performance changes due to virtually any change that may occur.

The lack of effective performance prediction tools is one of the reasons Capacity Planning is poorly executed in most shops. Without effective modeling tools, IT shops rely on rules of thumb to predict when they are out of capacity. The most common technique is to upgrade when processor utilization reaches a predefined limit. This may work well for some shops some of the time. But the link between processor utilization and workload performance is not clear cut. Certainly, the performance of some workloads degrades when total utilization nears 100%, but they are usually low priority work.

As workloads grow, and processor utilization nears 100%, this pattern develops:
• First, the lowest priority workloads begin to elongate, but higher priority work seems unaffected.
• As workloads continue to grow, utilization stays at or near 100%, but now medium priority work begins to suffer.
• Eventually the pain works its way up the priority chain till even the high priority work slows down.
Service Level Agreements (SLAs)

This scenario points out the difficulty of equating high processor utilization with an out of capacity situation. If we go back to our earlier definition for out of capacity, we need to look at the performance of each class of work, and to compare that against its service level requirement.

Another characteristic of performance is that it often follows an exponential curve. That means, as workloads grow, performance stays relatively constant until it reaches the point where it deteriorates suddenly and without much warning. This is called the knee of the curve effect and reflects the point on the performance graph when things go quickly from good to bad. If IT shops want to effectively manage their capacity and service delivery, they must be able to predict where they are on this curve, and especially, when they are approaching the knee of the curve. And that requires a modeling tool like Performance Modeler. 

"Capacity planning methodology” describes a methodology that uses Performance Modeler for capacity planning.

Capacity planning methodology

All shops have SLAs. If they are written down or stored electronically, they are explicit. But even when they are not formalized, IT shops and their users have an implicit understanding of where the line is drawn between acceptable and unacceptable performance. Sometimes the understanding is that performance for important workloads cannot get any worse than the current level of service. This certainly makes it hard to agree on when the line is crossed, but that won’t stop the phone from ringing when service degrades. SLAs are workload dependent. Important work usually has more stringent service requirements than lower priority work. That makes it essential that IT shops develop a baseline knowledge of current performance and that it be at the workload level.

The first step in developing a capacity plan is to segment all work into named workload groups. Throughout this manual, workloads have been segmented by either Performance Group Number (PGN) if the system runs in Compatibility Mode, or by Service Class Name if the system runs in Goal Mode. This scheme was chosen since it matches the way accounting data is collected and reported. But you can also create Reporting Groups and Reporting Classes if you need to classify work at another level. The following methodology assumes PGN or Service Class is an acceptable way to classify your workloads.

The Capacity Planning process begins by running RMF CPU and Workload Activity reports for your peak periods of processor utilization. This may be for one hour during prime shift, or may include one or more hours during third shift. These reports should be for the same hour (or multiple hours) each day, for 5 to 10 consecutive workdays. Each report should be for the default reporting interval, usually either 15 or 30 minutes in length. These reports provide the high level view of how the processor is being used and which workloads are the biggest consumers of resource. See Chapter 10, “Synergy with spreadsheets,” on page 61 for an explanation on how to transform these reports into meaningful charts. The next step involves building a baseline model which represents how the system runs during a typical peak period of utilization. Chapter 8, “Automatically generating a model with the Build option,” on page 45 showed how CPU and Workload Activity Reports could be generated by extracting RMF reports using the Extract function of Performance Modeler. The automated Build function can be used to scan these reports and let you build a model using the actual data in one report interval. The challenge is to pick an interval that represents a typical interval of high activity. Remember, this determines the baseline model and is used for all future comparisons. You should pick an interval of high activity. But if there
is a large variation between reporting intervals, you may want to pick an interval when activity was high, but not the highest activity. One technique is to examine all the reporting intervals (Step 4 in the Build process), and pick the interval when utilization was second or third highest.

Once the report interval is chosen, you must select the workloads for modeling. The workloads chosen for modeling should include the biggest consumers of resources as well as any workloads which have an SLA defined. These probably include most high priority workloads. But low priority work can also be modeled if you are interested in their performance. For example, test batch work may not have an SLA, but you may still be interested in seeing how this workload performs as workloads grow and processor utilization increases. Use the Build option to select the report interval and the workloads that are modeled. The result of the Build process is a set of modeling input parameters.

After the model is built you may have to alter some of the input parameters. For example, the Build process does not change paging rates from the default of zero paging. Nor does the Build process change the Number of I/O operations per online transaction from the default value (50 I/Os per tran). If you know the actual values for these parameters you can change them now, or you can leave them as they are. Paging rates are reported in RMF reports and can be readily obtained, but measuring I/O rates can be difficult. If you do not know the I/O rate and elect to use the default, you may see modeled response times that do not match the measured response times. But this can be corrected by adjusting the I/O rate or the disk response times and rerunning the model till the results match the measured values. In effect, this is another form of calibrating the model.

Whether you choose to calibrate the model till all response times are correct, or whether you accept some variations in the modeling results, the base model serves as a yardstick for all future comparisons. In fact, the I/O component of response times rarely determines when workload response times hit the knee of the performance curve. That is more a factor of how busy the processor is and the workload priority relative to other workloads. So accepting a variation in I/O times does not change the major conclusions of the modeling runs.

The Build process also chooses a default value for online path lengths. That default is 1 million instructions per transaction. This default makes it easy to calculate the transaction rate. As described earlier, transaction rate = MIPS consumed / path length. Once again, the path length and transaction rate may not match the real values for this workload. If you know the real transaction rate or path length, you can change these values. But in either case, the MIPS consumed by this workload during simulation matches the real MIPS consumed. And that is the most important result when modeling performance.

But I/O times and path lengths are important when modeling batch jobs. These parameters determine whether a batch job uses more or less CPU in a given interval. That is why the Calibration function exists. This function allows the model to automatically adjust the CPU to I/O ratio for each batch job until the correct amount of MIPS (from RMF reports) is consumed.

After adjustments are made, and calibrations are complete, we are ready to establish the baseline of performance metrics for each defined workload. These can be read from the modeling results report and imported into a spreadsheet. See Chapter 10, “Synergy with spreadsheets,” on page 61 for a detailed explanation of how modeling results can be easily imported and formatted in a spreadsheet.
Figure 22 shows an example of how the results could be displayed in a spreadsheet.

![Spreadsheet image](image-url)

**Figure 22. Simulation Results summary**

This table shows the baseline results on the first row, followed by modeling results for a 20% increase in workload volume. The processor for the baseline run is a 9672-R46. After showing results for the R46, the table shows what would happen if the R46 is upgraded to an R56 and then an R66.

The columns in this table include:
- **MIPS** Total MIPS rating for the R46 used in the model.
- **Total CPU % Busy** Total processor utilization according to the simulation report.
- **Model LPAR % Busy** Processor utilization consumed by PROD LPAR alone.
- **ONLNPROD, PDB2 and PEDASRV** Average response times according to simulation report.
- **BATCH 2 and BATCH 3** Average Elongation Factor for Batch workloads.

In this example, we wanted to see the impact on performance when workloads grow by 20% over current volumes. In addition to showing the modeled results, the spreadsheet was used to show the relative changes in each modeling run compared to the baseline run. These results are shown in the row labeled % change=>. In each run, the % change is measured against the baseline.

For example, the table shows that PDB2 response times increase by 18.42% if the workload volumes increase by 20% on the R46. The table also shows that each workload responds differently. In this example, the dispatching priority for these workloads are highest for workloads on the left and lowest for workloads on the right. This is reflected in the % change being greater for the workloads on the right. For example, BATCH 3 shows the largest relative increase in elapsed times compared to higher priority work.
These results show how to quantify the impact of change on performance, and to see the impact at the workload level. The next challenge is how to use these results in a capacity planning process. If you have SLAs in place, you can use the modeling results as a direct measure against your SLAs. But even without formal SLAs you can still use these results to reach conclusions on whether you have enough capacity. For example, you may go down the list of workloads being modeled and look at the % change for each workload. For each workload, you can ask the question, if response times grow by this percent, will performance be acceptable or not acceptable? Certainly for high priority work, a 500% increase in online response times is not acceptable. For low priority work, this may be acceptable. The benefit of this analysis is that it identifies whether workloads are to the left or right of the knee of the curve. For IT shops which do not have SLAs currently in place, this analysis may form the basis of creating SLAs.

Summary

The methodology described above can be summarized by these steps:

**Step 1** Gather RMF CPU and Workload Activity Reports till you identify periods of high utilization or times when capacity is constrained.

**Step 2** Select the workloads that are important and have performance requirements.

**Step 3** Build a baseline model using a representative reporting interval. This can be the interval of highest utilization, or an interval when utilization was nearly at its peak value.

**Step 4** Run the model to see how performance changes over time. You can model the impact of workload growth, or you can model the impact of configuration changes.

**Step 5** Summarize the results to show the absolute as well as relative change in performance for each modeled workload. Compare these results against your SLAs. If you do not have formal SLAs, use the relative change in performance to determine if the predicted performance is acceptable or not acceptable. As long as one workload is defined as having unacceptable performance, you are by definition out of capacity.

At this point you can use the model to investigate the impact of upgrading to a new processor with greater capacity.

Examples and tips

This section discusses how to model some of the most common workload environments. As with any tool, the value of Performance Modeler depends on how you use it. Although most input parameters are straight forward and easy to understand, some can be confusing or misunderstood. This section explains how to translate real configurations into the correct input parameters.

** Modeling TSO**

TSO workloads should be modeled as multi-tasking (Type=M) online workloads. Each TSO transaction runs independently from another TSO transaction. Most installations define multiple periods for TSO. As TSO transactions accumulate service they move from one period to another. And as they move, their priority (or Importance in Goal Mode) also changes. In the table we see Workload Activity Report information for three periods of TSO activity:
Modeling TSO

Workload Analysis for MVS1 on 7/8/99 at 11:45-12:00

<table>
<thead>
<tr>
<th>Workload</th>
<th>PGN</th>
<th>Period</th>
<th>Priority</th>
<th>CPU (%)</th>
<th>Total (%)</th>
<th>Cumulative MPL</th>
<th>Xact/Sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSO</td>
<td>1ST</td>
<td>PERIOD</td>
<td>1</td>
<td>93</td>
<td>1.40</td>
<td>0.16</td>
<td>0.00</td>
</tr>
<tr>
<td>TSO</td>
<td>2ND</td>
<td>PERIOD</td>
<td>2</td>
<td>91</td>
<td>0.60</td>
<td>0.07</td>
<td>0.00</td>
</tr>
<tr>
<td>TSO</td>
<td>3RD</td>
<td>PERIOD</td>
<td>3</td>
<td>90</td>
<td>3.90</td>
<td>0.43</td>
<td>0.00</td>
</tr>
</tbody>
</table>

In this example, TSO is running in Performance Group Number 8. As transactions migrate from period 1 to 2 to 3, their respective dispatching priority is lowered (93 to 91 to 90). The recommended technique for modeling TSO workload is to define three workloads. Each workload would represent a different period. The model priority assigned to period 1 should be highest and the priority for period 3 lowest. Since the transaction rates are readily available, you can use these values to break into the average path length for these workloads. For example, according to this table, TSO period 1 consumed .16% of this processor (R94 = 322 MIPS). This means .0016 x 322 or .51 MIPS were consumed. If the transaction rate was 1.32 transactions per second, then the average path length must be .51 MIPS / 1.32 = .390 million instructions per transaction. You can use this calculation to solve for the actual path lengths for all TSO periods.

Modeling CICS and IMS

Online workloads like CICS and IMS can be modeled in a number of ways. If you want to model the performance of a single CICS or IMS region they can be modeled as a single workload of Type=S. But most CICS regions do some degree of multi-tasking. Although most of the execution takes place under the single CICS main TCB (Task Control Block), some execution are run under separate TCBs. VSAM file I/O, for example, creates multiple TCBs under heavy load. And the Terminal I/O manager also runs under its own TCB. If we model all the CICS workload as a single TCB (Type=S), we would have a very conservative and possibly negative picture of what is really happening. The result could be a model that shows a single CPU constraint when it really doesn’t exist. Single CPU constraint occurs when a single tasking workload consumes near 100% of a single CPU.

One way to consider that some amount of CICS is running under a separate TCB is to break the workload into two copies. Each workload would be defined as Type=S, but the amount of CPU consumption would be different. For example, consider a single CICS region where we want to model 90% of the CPU running under one TCB, and 10% under a second TCB. Let’s also assume this region is consuming 50 MIPS capacity. One way to model this workload is to define two workloads CICSB (CICS Big) and CICSL (CICS Little). If we assume the same transaction rate for both workloads, we can control the MIPS consumed by assigning 90% of the average path length to CICSB and 10% to CICSL. Once again, a path length of 1 million instructions means we need a transaction rate of 50 trans per second to consume 50 MIPS. Thus we can assign 50 trans per second to both workloads but assign .9 x 1 = .9 million instructions as the path length to CICSB and .1 x 1 = .1 million instructions to CICSL. This gives a net MIPS consumed of 50 but allocate 90% of that to CICSB and 10% to CICSL.

In this example, we assumed that 90% of the CICS work ran under a single TCB. But if the CICS region uses DB2, the amount of multi-tasking may be much higher than 90/10. Since DB2 creates multiple TCBs, it may be appropriate to model this CICS region as a Type=M workload. That depends on how much CPU is consumed by DB2 versus the CPU running under the main TCB. This information may be difficult to determine and is beyond the scope of this manual. One rule of thumb for a DB2 intensive CICS region is to allocate half the MIPS to a
single tasking CICS workload and the other half to a multi-tasking (Type=M) DB2 workload. In the example above, if the CICS region is DB2 intensive, we could spread the workload over two workloads, CICS and DB2. The CICS workload would be defined as Type=S, the average path length set to 1 million instructions per second, and the transaction rate set to 25. This workload would consume 50% of the 50 MIPS. The DB2 workload would be defined as Type=M, with the same path length and transaction rate as CICS. This would consume the other half of the 50 MIPS but would spread the work over multiple TCBs.

**Modeling workload growth**

One of the common requirements for capacity planning is to model the impact of workload growth. Workloads grow for several reasons. Over time, the volume of work processed by existing applications tends to increase. This may reflect an increase in the amount of data being fed into an application, or it may reflect end users who are more efficient and faster at hitting the enter key. But experience has show that most mainframe applications grow in their use of resources over time. One of the more accurate ways to predict future growth is to look at historical trends. This is often called navigating by looking at the wake. But growth also occurs in sudden large increases as when a new application is installed or when a major rewrite occurs. A typical exercise for the capacity planner is to project these workload increases over time and model the impact on performance and capacity requirements. Performance Modeler includes two features that make it easy to model the impact of growth over a time interval. First, the Workload Definition screen contains a button (X) next to each workload Average Arrival Rate field. This is the multiplication button and provides you with a simple way to increase workload volumes. When the multiplication button is selected, you are asked to enter a multiplication factor. This factor is used to multiply the arrival rate. For example, a multiplication factor of 1.50 increases the arrival rate by 50%. That effectively increases the MIPS consumed by this workload by 50%. Using the multiplication button makes it easy to increase the amount of online work being modeled.

But batch workloads must be handled differently. The amount of work processed by batch workloads can be increased by raising the # of Users parameter. For batch workloads, this parameter controls the multi-programming level (MPL). For example, a value of two tells the simulator to create two identical batch jobs. For example, if the MPL for a batch workload is currently two, changing the MPL to three is equivalent to a 50% increase in that workload. Changing the batch workload MPL is preferable to changing other parameters, such as path length, since it keeps the CPU to I/O ratio unchanged. The MPL can be a fraction (non-integer) so you don’t have to worry about rounding the MPL up or down to the nearest integer.

Another feature that helps you to model time dependent changes is the Multiple Run option. This option lets you run several predefined model definitions in batch mode. For example, suppose you are interested in modeling the effects of a constant rate of growth for all workloads. And suppose you want to look at several points in time such as every six months over two years. You could build four model definitions, one for each six month interval, and store each set of definitions with its own file name. Next you could use the Multiple Run Option to batch all four models so that they run one behind the other without manual intervention. You can also request that each output report be written to disk, or printer, so you can see all the results in one place.
Latent demand in batch workloads

Latent demand in batch workloads

One of the most difficult exercises for the capacity planner is to estimate the amount of latent demand in their system. This is the amount of work that is not getting done because of resource constraints. Latent demand can be viewed as pent up demand, waiting for additional capacity so it can break out. Understanding latent demand is important since it determines how systems perform when new capacity is added. Consider this scenario:

• The old system runs at 95% to 100% processor utilization during peak hour of the day.
• The processor is replaced with a new processor rated as two times the capacity of the current machine.
• The expectation was that processor utilization would drop to 50% to 60%. In fact, processor utilization only drops to 80%.

The reason that utilization did not drop as low as expected is that a large amount of work was not able to run on the old machine, because of the high utilization. When capacity is added, this work is able to use the additional capacity and drive utilization to higher than expected levels. This is a common scenario. Most processors are not upgraded until they begin to run at high utilization. That means there are probably workloads running at low priority that are being throttled or held back. Since batch workloads are more likely to run at low priority (compared to online workloads), these are the workloads which account for most examples of latent demand. The following section discusses how Performance Modeler can be used to measure and quantify the effect of batch latent demand.

Batch workloads differ from online workloads in the way they consume CPU resources. The total MIPS consumed by an online workload can be calculated by multiplying the transaction rate by the path length (transactions per second x millions of instructions per transaction = millions of instructions per second or MIPS). This relationship holds as long as there is sufficient capacity to allow the online workload to run at the specified transaction rate. High priority online workloads rarely demonstrate latent demand. But batch jobs consume as much CPU capacity as they can. Batch jobs are only limited to the extent that they are I/O bound and by how often they are preempted by higher priority jobs. Batch jobs that run at low priority are often waiting while higher priority work use CPU resources. The higher the processor utilization, the more time these jobs spend waiting for a free CPU. The time waiting is called Queue time. Queue time is directly affected by total processor utilization. But Queue time is also affected by other constraints. For example, LPAR definitions may limit the capacity available to batch jobs even when total utilization is below 90%. But these definitions and their effects on batch performance can be modeled using Performance Modeler.

When additional capacity is modeled, the largest impact is usually seen by the lowest priority workloads. When these are batch workloads, the result is a reduction in the modeled Elongation Factor for these workloads. Since Elongation Factor is a direct measure of batch elapsed times, any reduction in Elongation Factor is accompanied by an increase in processor utilization. In effect, batch workloads are running faster and ending in less time by consuming more MIPS. The amount of increase in MIPS consumed depends on how constrained were these workloads before capacity was added. The model is showing the impact of reducing the Queue time and the corresponding increase in utilization. This increase in utilization can be defined as batch latent demand.

One word of caution on how to interpret these results. Batch workloads are modeled as never ending tasks. That means each batch job continues to run for the
total duration of the model. If you model five batch jobs within a batch workload (MPL=5), these five jobs keep running for the length of the model run. When you model additional capacity, these five jobs run faster (lower Elongation Factor), but the MPL is kept constant at five. In real life, when batch jobs end sooner, the average MPL may actually go down. But the model assumes there is an endless supply of batch work to maintain the same MPL, no matter how fast these jobs run. That means the model depicts a worst case scenario regarding latent demand. The latent demand shown by the model is probably on the high side. But that depends on how much batch work is waiting to run. If you believe better batch performance causes the average MPL to go down you can model that too. For example, change the MPL from five to four, or whatever makes sense for your installation. If you want a worst case view, or believe the batch MPL does not change, leave it at five.
Chapter 10. Synergy with spreadsheets

Spreadsheet programs such as Lotus® 1-2-3® and EXCEL are excellent tools for converting numeric data into presentation material. The ability to create charts and annotate tabular data make these programs powerful tools for converting the data generated by Performance Modeler into final presentations. In order to exploit these capabilities, many of the output files generated by Performance Modeler can be written to disk in CSV (Comma Separated Value) format. That makes it easy to import these files into a spreadsheet and store each data point in its own cell. That also makes it possible to use the charting capabilities of 1-2-3 and EXCEL to convert these numbers into easy to read graphs.

After you select the Performance Modeler print option to write out results you see a form that asks if the results should be written to a printer or to a disk file. Disk output can be "text" or "numeric" or both. Text output writes the file as a text file and it looks the same on disk as if the file was written to a printer. Numeric output writes out a smaller version of the output report, but more important, the data is in CSV format. That makes it easy to import this file into a spreadsheet for later manipulation.

One of the tools included in the Performance Modeler package are two spreadsheets. One is called CPFPM.123 and is designed to be used with Lotus 1-2-3 Millennium Edition (or later version). The other is called CPFPM.XLS and is designed for use with Microsoft EXCEL. These spreadsheets were designed to make it easy to import the data generated by Performance Modeler, and to quickly create meaningful charts. An example of the 1-2-3 spreadsheet is shown on the following page.
How to create a chart

The CPFPM.123 file contains four spreadsheets. The first sheet labeled INPUT contains most of the input fields that allow you to specify which files should be imported and how they are stored in the spreadsheet. For example, the fields in blue are input values, the command buttons to the right contain LOTUS Scripts (a programming language) which automatically import and graph the files specified in blue.

The Directory field specifies the location (default disk directory) which is searched for all the files defined below. The directions in red (on the right) are meant to guide you through a step by step process to build a capacity planning presentation. These steps should be followed in the order shown. For example, if you select the Step 4 button before selecting the Step 1 button, Performance Modeler displays an error message.

Here is a description of these steps and their functions.

Figure 23. Lotus 1-2-3 spreadsheet file
Step 1 Import the LPAR utilization report

This button imports the LPAR CPU Utilization data which was created when the RMF/CMF extraction function was run. This is the report created when the LPAR report was extracted and stored on disk. This file is usually stored with a file name LPAR.RPT.

Since the file was stored in CSV format, the results are imported into the spreadsheet with each data point stored in its own cell. The LPAR Data Sheet Name input field lets you name the sheet where the data is stored. The LPAR Data File Name is where you specify the name of the extracted report you want to import. Once these fields are filled in, you can select the command button (LPAR Data Button). When selected, the scripting language behind the button imports the extracted file and stores it in the specified sheet. In this example, the file name LPAR.RPT is imported into a sheet with name CPUR26. If a sheet with this name already exists, it is erased and recreated with the imported data.

An example of the new sheet, after the report is imported, is shown in Figure 24.

The new sheet labeled CPUR46 now contains the imported LPAR.RPT file. In addition to the CPU utilization data, it contains information about the size and dimensions of the data table. This information, shown at the top of the sheet, is used when the data is graphed in subsequent steps.

Figure 24. Imported data
Step 1 Import the LPAR utilization report

Starting in row 5, this table shows the utilization of an R46 running two LPARs, PROD and TEST. Each row shows the results for a different reporting interval.

Step 2 Generate the LPAR utilization graph

This step generates a CPU utilization graph based on the data imported in Step 1. The graph is stored in a new sheet with the name entered in the LPAR Graph Sheet Name field. The first two title lines in the graph are taken from the fields LPAR Graph Title Line 1 & 2.

An example of the graph created by selecting the LPAR Graph command button is shown in Figure 25.

![Image of the CPU utilization graph](image)

Figure 25. The CPU utilization graph

The graph is created as a combination line and stacked bar chart. The black line at the top shows total utilization for the entire processor. The stacked bars show the utilization for each individual LPAR. In this example, CPU utilization for LPARs PROD and TEST are shown.

Step 3 Import and sort the workload

This button imports the file created when the RMF/CMF Utilization Report Workload Reports are extracted. In addition to importing this data, the results are sorted so they can be graphed (Step 4) by individual workload name.
In this example, the results are imported from the file WGL.RPT and stored in a new sheet named WGLR46.

An example of this new sheet, after the report is imported, is shown in Figure 26.

The new sheet named WGLR46 now contains the workload utilization data, sorted by Service Class name, and sorted by Priority. This data is the input to Step 4 where the workload utilization graph is created.

**Step 4 Generate the workload**

This button generates the Workload Utilization Graph showing Utilization Graph utilization at the workload level. If the system runs in Compatibility Mode, the workload is at the period within Performance Group Number. If the system runs in Goal Mode, the workload is at the period within Service Class level of detail.

The input to this graph comes from two places. The Total and LPAR utilization data is read from the sheet created by Step 1 (LPAR Data Sheet). The second set of input comes from the sheet created by Step 3 (Workload Data Sheet). Similar to Step 2, the first two title lines for the new graph come from the input title fields in Step 4.

An example of the graph created in Step 4 is shown in Figure 27.
Step 4 Generate the workload

The black line shows the Total Utilization for the entire processor. The red line shows the utilization for the PROD LPAR. The stacked bars show the utilization for each workload. Since the workloads are also sorted by priority, the highest priority workloads are plotted first (top of the legend, bottom of the actual bars). That makes it easy to see how high priority work can sometimes squeeze out the lower priority workloads. In this example, the lowest priority workload is BATCH_3. When the processor runs at or near 100% busy, this workload is bounded on the top by the red line, and bounded on the bottom by higher priority work within the LPAR.

One of the options provided in the INPUT sheet for STEP 4 is the field labeled Minimum % Busy for Graphing. This sets the minimum value for CPU utilization for selecting a workload for graphing. In this example, any workload which used at least 2% of the processor, in any reporting interval, is selected for graphing. This lets you limit the number of workloads that are graphed to a manageable number. It also makes the graph less busy and easier to read. But this also leads to some of the white space that appears between the red line and the top of the stacked bars. This white space represents the utilization within the LPAR which is not shown broken out by workload. It is made up of those workloads which use a small amount of CPU (less than 2% in this example) plus the Uncaptured Time. Uncaptured Time is the actual utilization which cannot be charged back to an individual workload. Typically, Uncaptured Time is approximately 7%-10% of the actual utilization of the entire LPAR.
Step 4 Generate the workload

One important note to remember when selecting the Step 4 Command button is the relationship between the graph it creates and the data in the LPAR Data Sheet. At the top of the LPAR Data Sheet is a field labeled - LPAR Column. By default, the LPAR Column is set to D (see Cell G1). This value designates the column which shows the CPU Utilization of the first LPAR found in the LPAR Data Report. But it also is used to create the Red Line in the Workload Utilization Graph. In this example, the Workload Utilization Graph was created for the PROD LPAR. Since the PROD LPAR was in fact stored in the D column, the right utilization (red line) was created. If you want to create a Workload Utilization Graph for a different LPAR, you must change the LPAR Column to the proper value. For example, if you wanted to create a Workload Utilization Graph for the TEST LPAR you must change the LPAR Column from D to E in the LPAR Data Sheet. This must be changed before Step 4 is selected.

Occasionally, the Workload Utilization Graph does not look right. The most common problem is the bars that make up the workload utilization values do not track with the red line (LPAR utilization). This problem is usually due to a mismatch between the LPAR Utilization Report and the Workload Utilization Report. This mismatch can occur when the number of reporting intervals is different between both reports. The Macro which creates this graph expects that both reports have identical reporting intervals. If you see some inconsistencies with this graph, check the reporting intervals to make sure they are the same in both reports.

Step 5 Import the workload analysis table

This button imports a Workload Analysis Table showing CPU utilization plus other data for a specific reporting interval. This step lets you create a table in a spreadsheet which looks exactly like the workload analysis table displayed during the BUILD process.

In fact, before you can execute this command you must first go through the BUILD process. Step 5 of the BUILD process displays a form that shows a Workload Analysis Table for the selected reporting interval. If you select the PRINT option, and then select the option to write a numeric disk file, a file called WKL.NUM is written to disk. This is the file which is imported when you select the Step 5 Command Button.

An example of a Workload Analysis Table created by Step 5 is shown in Figure 28.
Step 5 Import the workload analysis table

This table is stored in the sheet named in the Workload Table Sheet Name input field. In this example, it was imported from the file named WKL.NUM. One of the options you can specify in the INPUT sheet is the Minimum % Busy for Bold value. In this example it is set to 2%. That specifies that you want to Bold Type the workloads which used 2% or more of the processor. That makes it easy to see the workloads that consume the most CPU resources.

Step 6 and Step 7 Import and graph CPU utilization

The last two Steps, Step 6 and Step 7, are similar to Steps 4 and 5. They allow you to create a utilization graph at the workload level of detail. But instead of importing and graphing workloads based on Service Class Names, these steps import and graph CPU utilization based on Reporting Class Names.

Long term trending

The LPAR and Workload graphs are good ways to represent current CPU usage. A weeks worth of reports, showing the peak hour for each day, broken into 15 or 30 minute reporting intervals is the recommended duration for creating these graphs. They give you a good view of how the processor is being used, but don’t contain so much data as to muddy the graphs. If you run these reports once a week, you can build up a collection of utilization graphs, all stored within the same spreadsheet file, or stored in different files. By changing the input fields and executing the command buttons multiple times, you can build multiple graphs for
different reporting intervals. One of the benefits of using a spreadsheet to hold this data is the flexibility it affords you in how you want to present the data. You can combine multiple weeks of utilization data in order to show a month or several months worth of data.

This makes it easy to spot trends in growth or changes in the workload profile. That is an important prerequisite to capacity planning. Or you can change the graph options and quickly convert from a Stacked Bar chart to a Line only chart. And when the graphs and tables are in final format, you can convert them into PDF format, ready for presentation or distribution through the Internet. Due to the synergy between Performance Modeler and spreadsheet programs like 1-2-3, you may find you no longer need other report generation tools such as SAS or MICS for capacity planning.

Importing Performance Modeler results

Another function provided by the Performance Modeler spreadsheet is the ability to import the modeling results generated by one or more model runs. The SIMT and SIMN sheets contain command buttons that import the modeling results. The command button in SIMT imports the text results from one or more modeling runs. The button in SIMN imports the numeric (CSV format) results from one or more modeling runs. For these buttons to work, the results would have to be written to disk after each modeling run. For example, Figure 29 shows the results of selecting the Load SIM.NUM button in the SIMN sheet.
Importing modeling results

![Image of a spreadsheet showing the results of running three models, back to back, and writing the results to a file named SIM.NUM. The file was written in numeric format. When the Load SIM.NUM button was selected, the results were imported into this sheet, and each field stored in its own cell. If you want to import another set of results into this sheet, you can simply change the starting row value (currently set to 5) and select the Load button again.

These results can be converted into a table in final presentation format by using the command button in the next sheet (SIMSUM). The results of selecting the Load Results Summary button in the SIMSUM sheet are shown in Figure 30.

Figure 29. The SIMN sheet

This table contains the results of running three models, back to back, and writing the results to a file named SIM.NUM. The file was written in numeric format. When the Load SIM.NUM button was selected, the results were imported into this sheet, and each field stored in its own cell. If you want to import another set of results into this sheet, you can simply change the starting row value (currently set to 5) and select the Load button again.

These results can be converted into a table in final presentation format by using the command button in the next sheet (SIMSUM). The results of selecting the Load Results Summary button in the SIMSUM sheet are shown in Figure 30.
In this example, when the Load Results Summary command button was selected, several events took place. First, the title field in cell C1 was written to the top of the output fields, beginning on row 5. Row 5 was chosen as the start of the output based on the Start Row input field shown in cell C2. The rest of the output was created by copying the modeling results from the SIMN sheet, starting at row 5 and continuing to row 21. These are specified as input values in cells C3, E3, and G3 (see figure above).

Once the modeling results were copied, the command button also formatted the results to show the relative changes, comparing the first set of modeling results (Base Run on 9672-R46) to the following sets of results. The end result of these operations is a table that clearly shows the baseline run results compared to the other results, and easily identifies the relative changes in capacity and performance. This table, from row 5 through row 19, can be written to hard copy, or converted to PDF format for final presentation. By altering the different input fields, you can easily build a series of presentation tables, each standing separately, or as a group of results.
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
522 South Road
Poughkeepsie New York 12601-5400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

CICS          MVS          Tivoli
DB2           OS/390       z/OS
IBM           RMF
IMS           S/390

Trademarks owned by Lotus Development Corporation

1-2-3          Lotus

Trademarks owned by Microsoft Corporation

Microsoft      Windows

Other company, product, and service names may be trademarks or service marks of others.
Index

A
Average Arrival Rate 3
Average Path Length 3

B
baseline model 16
batch workloads 2
calibrating latent demand 58
modeling 3
Build option 7
Build process defaults 49

C
calibrating batch workloads 37
Calibration Run option 15
capacity planning 51
elements 55
methodology 52
tips 55
CICS
modeling 56
configuration definition screen 9
Configuration option 7
creating spreadsheet chart 62

defaults
Build process 49
defining fill in workloads 30
processor configuration 27

E
Edit option 7
derror messages from Extract process 43
Extract option 7
extracting RMF report data 41

F
File option 7
fill in workloads defining 30

G
generating (continued)
workload 65
generating (continued)
workload 65
generating (continued)
workload 65
getting started 27
graph CPU utilization 68
graphing workload performance 21

calibrating batch workloads 37
Calibration Run option 15
capacity planning 51
elements 55
methodology 52
tips 55
CICS modeling 56
configuration definition screen 9
Configuration option 7
creating spreadsheet chart 62

defaults
Build process 49
defining fill in workloads 30
processor configuration 27

E
Edit option 7
derror messages from Extract process 43
Extract option 7
extracting RMF report data 41

F
File option 7
fill in workloads defining 30

G
generating LPAR utilization graph 64

O
online workloads modeling 3
output described 2

P
Performance Modeler purpose 1
performance modeling basics 27
performance tracking 23
primary menu 7
Print option 15
Print Option screen 25
print options 24
processor configuration defining 27
MIPS discussed 27

R
reports CPU 45
Workload 45
Restart option 45
Return option 15
RMF report data extracting 41
Run option 7
run options 15
Run Wizard 16
Run Wizard option 15
running the simulator 15

S
Service Level Agreements 51
simulation models 1
simulation techniques 3
simulator running 15
Simulator Running screen 19
Single Run option 15
single tasking workloads 2
sorting workloads 64
spreadsheet chart creating 62
spreadsheet file producing with print option 25
spreadsheets 61
system requirements 5

T
text image 25
V
View option 8

W
workload definition screen 11
workload growth
modeling 57
workload performance 23
  graphing 21
workload types 2
workloads
  batch 2
  fill in defining
    defining 30
  importing 64
  multi-tasking 2
  selecting for modeling 29
  single tasking 2
  sorting 64
Workloads option 7
Readers’ Comments — We’d Like to Hear from You

Tivoli Performance Modeler for z/OS
User’s Guide
Version 2 Release 1

Publication No. SC31-6385-01

Overall, how satisfied are you with the information in this book?

<table>
<thead>
<tr>
<th></th>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall satisfaction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How satisfied are you that the information in this book is:

<table>
<thead>
<tr>
<th></th>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accurate</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complete</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy to find</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy to understand</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Well organized</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applicable to your tasks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please tell us how we can improve this book:

Thank you for your responses. May we contact you?  □ Yes  □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

_________________________        ___________________________
Name                                      Address

_________________________        ___________________________
Company or Organization

_________________________
Phone No.
Program Number: 5698-A18

Printed in USA