HP Performance Collection Software User's Manual

(for MPE Systems)

HEWLETT PACKARD

HP Part No. 50700-90022
Printed in U.S.A. 0492

E0492

Notice

Hewlett-Packard Company
SWT/Software Technology Center
8010 Foothills Boulevard
Roseville, CA 95678 U.S.A.

Printing History

Updates are new editions or complete revisions of the manual.

A software code may be printed before the date; this indicates the version level of the software product at the time the manual was issued. Many product updates and fixes do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.

Edition 1	3/91	50700-90022	E0391
Edition 2	4/92	50700-90022	E0492

Edition 1 of this manual was titled *HP LaserRX/MPE User's Manual: Collection Software*

This manual and the *HP LaserRX/MPE User's Manual: Analysis Software* replace part numbers 50700-90001, 50700-90010, and 50700-90016.

Introduction to Performance Collection Software

The HP Performance Collection Software captures performance data from HP 3000 MPE V and MPE/iX computer systems. It logs this data automatically and continuously while consuming only minimal system resources. The rich detail of the collected data allows for short term performance diagnosis while the compact storage allows long term trending from a single data source.

The HP LaserRX/MPE product provides a flexible, high powered, graphical user interface to examine the data captured by the Performance Collection Software. This companion product to the Performance Collection Software must be purchased separately and is not required in order to analyze the performance data. HP LaserRX/MPE does simplify the task of analyzing the performance data significantly with its top down approach to system performance and is highly recommended as the best way to view Performance Collection Software data.

The HP Performance Collection Software allows you to:

- Collect comprehensive performance data continuously.
- Analyze resource usage and isolate bottlenecks.
- Document findings and conclusions.
- Solve system management problems before they arise.

Related Documents

User information for HP LaserRX/MPE consists of the data analysis and display documentation included with your PC software and the Performance Collection Software manual included with your host software. Each manual is described briefly below:

For HP LaserRX/MPE users:

- *HP LaserRX/MPE User's Manual: Analysis Software* describes how to analyze performance data using your PC.[1]
- *HP Performance Collection Software User's Manual (for MPE Systems)* describes the HP 3000 software. Refer to it for information about the HP 3000 components of HP LaserRX/MPE.
- *HP LaserRX/MPE: A Journey of Discovery* guides you through performance analysis using the HP LaserRX/MPE performance tool.[1]
- *Using Basic Serial Connection Files* shows you how to create special command files for serial connections.[1]

For HP GlancePlus Pak users:

- *HP GlancePlus User's Manual (for MPE/iX Systems)* and *HP GlancePlus User's Manual (for MPE V Systems)* describe the GlancePlus diagnostic tool that is a companion product to Performance Collection Software.
- *HP Performance Collection Software User's Manual (for MPE Systems)* describes the HP 3000 software. Refer to it for information about the HP 3000 components of HP LaserRX/MPE.

[1] Supplied only with the HP LaserRX/MPE Analysis Software product.

More about the Collection Software User's Manual

The HP Performance Collection Software has three components: SCOPE (or SCOPEXL, if you are using an MPE/iX system), UTILITY, and EXTRACT.

SCOPE(XL) is the performance data collector for MPE Systems. It continuously collects and summarizes performance data.

UTILITY and EXTRACT are the host programs that let you interact with SCOPE(XL) and manage the data that it collects.

This document—the *HP Performance Collection Software User's Manual (for MPE Systems)*—describes how to use the collection and management software that runs on the host HP 3000 system. It describes how the host components interact, provides detailed command descriptions for each program, and suggests how to use the programs to analyze and archive data efficiently.

Note This manual is written for use with MPE V *and* MPE/iX systems. Where performance data collection information pertains only to MPE V systems, the components are named SCOPE and SCOPE2. Where data collection information pertains only to MPE/iX systems, the components are named SCOPEXL and SCOPEXL2. If the information pertains to either or both systems, the names SCOPE(XL) and SCOPE(XL)2 are used.

Conventions Used in this Manual

`Computer font`	Represents screen text, including prompts and messages appearing on the screen.
Italics	Identifies variable parameter values in command syntax statements.
UPPERCASE	Identifies commands and parameters that are to be entered exactly as shown in command syntax statements. It also identifies the names of reference programs, commands, and files. Uppercase is also used to reference program, command, and file names.
$\boxed{}$	Indicates the key on the terminal or PC keyboard represented by what is enclosed. For example, Press $\boxed{\text{Y}}$ directs you to press the "Y" key.
$\boxed{\text{Return}}$ *or* $\boxed{\text{Enter}}$	Represents the terminal *or* PC key used to execute a command.

Within Syntax Statements

{ }	Braces enclose required elements. When several elements are stacked within braces, you must select one.
[ ]	Brackets enclose optional items. When several elements are stacked within brackets, you can select any one element or none.
Punctuation	Except for braces, brackets, and ellipses, all punctuation characters must be entered exactly as shown.
[. . .]	Horizontal ellipses enclosed in brackets indicate you can repeatedly select elements that appear within the immediately preceding pair or braces or brackets. If a punctuation character precedes the ellipsis, you must use that character as a delimiter to separate repeated elements unless only one element is selected.

Using This Manual

This manual is designed to help you begin to use the Performance Collection Software and to support your continued use.

Chapter 1	*Installing HP Performance Collection Software* Describes hardware and software requirements, and explains how to install or update Performance Collection Software on your host system.
Chapter 2	*Performance Collection Software—An Overview* Introduces the capabilities of the four Performance Collection Software host programs—SCOPE(XL), SCOPE(XL)2, UTILITY, and EXTRACT—and discusses the software's functionality.
Chapter 3	*Data Collection: SCOPE* Provides detailed descriptions of the SCOPE(XL) PARM file directives and their parameters.
Chapter 4	*Data Management: UTILITY* Provides detailed descriptions of the UTILITY program commands and their parameters.
Chapter 5	*Data Management: EXTRACT* Provides detailed descriptions of the EXTRACT program commands and their parameters.
Chapter 6	*Archiving Strategies* Gives recommended Performance Collection Software data archiving strategies.
Chapter 7	*Performance Collection Software Performance Alarms* Provides a detailed description of the Performance Alarm feature of the SCOPE and SCOPEXL performance data collection programs.

Within Syntax Statements

Chapter 8	*What's New—Changes to Performance Collection Software*
	Describes changes made to the software since version A of the HP LaserRX/MPE release.
Appendix A	*MPE/iX Metric Information*
	Gives additional metric information for MPE/iX users.
Appendix B	*Service Level Agreements*
	Describes the Service Level (SLA) enhancement to the SCOPEXL performance data collection program.
Appendix C	*UTILITY SCAN Report Details*
	Describes various reports that you can generate with the UTILITY SCAN command.
Glossary	Provides a quick, alphabetized reference to terms used in this manual.

1

Installing HP Performance Collection Software

Introduction

This chapter tells you how to install the HP Performance Collection Software on an HP 3000 MPE V or MPE/iX system.

Whether you are a new or existing user of HP Performance Collection Software you should follow these instructions to install or update Performance Collection Software on your HP 3000 system (or systems).

What You Will Need

This section describes the hardware and software requirements for the HP 3000 system.

Hardware Requirements

The HP 3000 system (or systems) from which you will collect performance data for analysis must be equipped with the following:

- Either of the following, if the HP LaserRX/MPE analysis software is used:

  ∗ An RS-232 asynchronous interface (for basic serial connections).
  or
  ∗ A LANIC and ThinMAU interface (for Local Area Network connections).

- Approximately 60 MB of disc storage for Performance Collection Software log files.

Within Syntax Statements

Software Requirements

To be able to run Performance Collection Software, your HP 3000 must have the following software installed:

- One of the following:

  ∗ MPE V G.03.02 (V-Delta-2 MIT) or later (for MPE V systems).
  or
  ∗ MPE/iX release A.41.00 (2.1) or later (for MPE/iX systems).

- Local Area Network software, if you use this type of network connection when running the HP LaserRX/MPE analysis software.

Configuration Requirements

On MPE V systems, the following values must be configured:

- Set the MAX CODE SEGMENT SIZE to its maximum of 16384.
- Set the MAX # OF SEGMENTS/PROCESS to 16 (or more).
- Set the MAX STACK SIZE to its maximum of 31232.
- Set the MAX EXTRA DATA SEGMENT SIZE to its maximum of 32764.
- Set the MAX # OF EXTRA DATA SEGMENTS/PROCESS to 4 (or more).

See the *MPE V System Manager/System Supervisor Reference Manual* for further information on changing these values.

Installing or Updating Performance Collection Software

Starting with MPE/iX release B.40.00 (4.0) and MPE V release G.23.00 (23), the Performance Collection Software is installed as part of the normal operating system update process.

Installation Instructions for first time installations

1. Consider Security and Accounting requirements.

   Performance Collection Software program files are installed in the SCOPE.SYS group. This group requires a minimum capability set of "CAP=IA,BA,PH,DS,MR,PM". Access to the SCOPE.SYS group will normally allow any user to execute programs, but only users logged on to the group (or users with SM capability) will have any other access to its files.

   The SCOPE.SYS user will be created with normal user capabilities and the SCOPE.SYS group as the home group.

   Since the SCOPE.SYS group requires PM capability, and since system passwords may be added to job streams in this group, you should add unique passwords to the SCOPE.SYS group and SCOPE.SYS user. For example:

   ```
   :ALTGROUP SCOPE;PASS=grppass
   :ALTUSER  SCOPE;PASS=userpass
   ```

2. Add your system identifier to the PARM file.

   The system identifier is used to identify log files created on this system. The first ten characters of this identifier are used as a title by the HP LaserRX/MPE Analysis software. You should edit the file PARM.SCOPE.SYS to include a unique system identifier for each of your systems.

Within Syntax Statements

A system identification record may occur anywhere in the file but there should be only one such record. Any standard text editor may be used to edit the PARM file as long as the final result is a standard ASCII file with records less than or equal to 100 characters wide.

Example:

```
:EDITOR
TEXT PARM.SCOPE.SYS
ADD .1
ID=HARVEY
//
KEEP PARM.SCOPE.SYS,unn
```

You may check the validity of the PARM file if desired:

```
:RUN UTILITY.SCOPE.SYS
PARMFILE PARM.SCOPE.SYS
(any errors and the parm file contents will be listed)
EXIT
```

3. Add any required passwords to the SCOPEJOB.SCOPE.SYS job stream file. It should log on as MANAGER.SYS,SCOPE or any other user which has SM capability. The performance log files will be created in the logon group for this job.
4. Start the data collection.

   ```
   :STREAM SCOPEJOB.SCOPE.SYS
   ```

5. Arrange to have the data collection started each time the system is rebooted. The easiest way to do this is to add the following line to the SYSSTART.PUB.SYS file.

   ```
   :STREAM SCOPEJOB.SCOPE.SYS
   ```

Update Instructions

These instructions are for installations where previous versions of the Performance Collection Software or HP LaserRX/MPE host software were installed.

- If the previous release was B.00.00 or later then no further action is required in order to complete the update process.
- If the previous release was A.00.00 - A.99.99 then your existing log files must be converted to the "B" release format before they can be used. At a minimum you must convert the raw log files before you start the data collection. Extracted files may be converted any time before they are used.

Steps for updating from the "A" to the "B" release software:

1. Store all raw and extracted log files to tape.

```
:STORE @.SCOPE.SYS;*T
```

2. Run the UTILITY program to convert the raw log files.

```
:HELLO MANAGER.SYS,SCOPE
:RUN UTILITY.SCOPE.SYS
CONVERT LOGGLOB
EXIT
```

3. You may convert any extracted log files at this time or wait until later if desired. To convert extracted log files, run the UTILITY program and execute the CONVERT command for each one. For example:

```
:RUN UTILITY.SCOPE.SYS
CONVERT file1
CONVERT file2
```

Within Syntax Statements

4. Examine the SCOPEJOB.SCOPE.SYS job stream and insure it has a CONTINUE command prior to running the SCOPE or SCOPEXL program. This insures the rest of the job stream will be processed even if the collection software terminates abnormally.

   For MPE/iX systems:

   ```
   :
   !CONTINUE
   !RUN SCOPEXL.SCOPE.SYS
   :
   ```

   For MPE V systems:

   ```
   :
   !CONTINUE
   !RUN SCOPE.SCOPE.SYS
   :
   ```

5. Restart the performance collection software.

   ```
   :STREAM SCOPEJOB.SCOPE.SYS
   ```

If You Need Additional Help

Resolving Incompatible Measurement Interface Versions

If you try to start collecting data by running the SCOPEXL program, but you have installed the incorrect Performance Collection Software files on the host, you will get an error message similar to the following:

```
******************************************************
Potentially incompatible Measurement Interface versions
exist between Scope/XL and this system.

This is Scope/XL version B.06.02

This system is using Measurement Interface version: X.07.01.

Please contact your local Hewlett-Packard
Support Representative for resolution.
******************************************************
```

This means that the version of the SCOPEXL program you are using is not compatible with the version of the operating system. Check to make sure you have installed or updated your performance collection software from the proper source.

If the installation procedures do not complete correctly after you have followed the steps listed in this chapter, call your Hewlett-Packard support representative for assistance. Please have the following information available when you call:

- The version number of your Performance Collection Software

  ```
  :RUN UTILITY.SCOPE.SYS;INFO="VERSIONS"
  ```

- The version number of the MPE operating system on the HP 3000

  ```
  :SHOWME
  ```

- The exact text of any error messages encountered.

2

Performance Collection Software An Overview

Introduction

Performance Collection Software consists of four basic programs. This chapter discusses how these programs interact, describes the modes in which Performance Collection Software functions, and provides information about softkeys and native language support (special features of the UTILITY and EXTRACT programs).

The basic programs are

- SCOPE (or SCOPEXL, depending on your system),
- SCOPE2 (or SCOPEXL2),
- UTILITY, and
- EXTRACT.

Performance Collection Software Schematic

The following schematic diagram shows the relationship between the different components of the Performance Collection Software system. The components comprising the host programs are shaded.

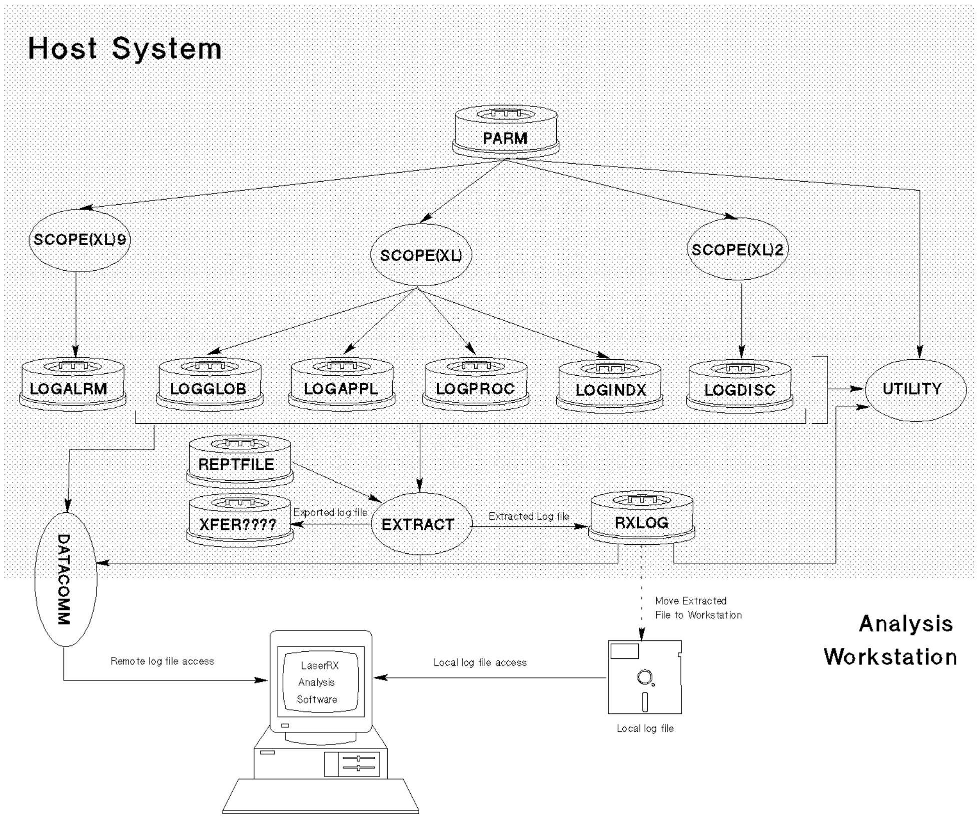

Figure 2-1. Schematic of Performance Collection Software Components

The Host component contains the following programs and files:

SCOPE *or* SCOPEXL	Main performance data-collector program.
SCOPE2 *or* SCOPEXL2	Disc space data-collector program.
EXTRACT	Program to extract selected data from log files.
UTILITY	Program to perform various utility functions.
DATACOMM	Library code to allow work stations to access host log files.
PARM file	User-generated parameter file that controls data collection.
LOG ...	Log files that contain raw performance data.
Extracted LOG files	Log files that contain selected performance data. This file can be accessed on the host or moved to the HP LaserRX/MPE PC analysis workstation for local access.

Current HP LaserRX/MPE Users

Significant changes were made to enhance the Performance Collection Software product since the version-A releases (versions A.*nn.nn*) of HP LaserRX/MPE. These changes include the following:

- New log file formats.
- A command-driven user interface for UTILITY and EXTRACT.
- The ability to log disc-space metrics.
- New THRESHOLD directives.
- Increased and enhanced functionality for UTILITY and EXTRACT.
- New log files in the SCOPE.SYS group.

For details on Performance Collection Software enhancements, *see* chapter 8.

Within Syntax Statements

Caution The command-driven user interface and the new log file formats require that you alter your UTILITY and EXTRACT batch jobs, and that you convert existing log files to the new format. For these reasons, we strongly recommend that you review the information on the UTILITY and EXTRACT programs before you try to use the new versions.

The operation of SCOPE(XL) has not changed. The batch job that runs the data collector should not require any changes.

About Performance Collection Software Host Programs

Based on the information the PARM file provides to SCOPE(XL), the collector program collects and summarizes system performance metrics, and writes these metrics to raw **log files**. Because SCOPE(XL) runs continuously, the UTILITY program provides a special set of commands to let you communicate with SCOPE(XL) while it is collecting data.

You can start and stop SCOPE(XL), add notes to the information being logged, and get a status report on the SCOPE(XL) program. You can also tell SCOPE(XL) to reread the PARM file to allow you to make PARM file changes without stopping the program. UTILITY commands let you open, create, resize, convert, and generate reports on raw and extracted log files. (Extracted log files contain subsets of the information included in raw log files. Use the EXTRACT program to generate extracted log files.)

EXTRACT lets you extract selected Performance Collection Software data from raw or extracted log files. You can extract data based on a variety of commands and parameters. These commands and parameters are described in chapter 5.

You should not run the SCOPE(XL)2 program directly. The SCOPE(XL) program will run it automatically, as necessary. The only exception to this rule is the need to run SCOPE(XL)2 with ;INFO="HELP" while generating the Disc Space Grouping report. For more details, see the section on Data Collection in this manual.

Interactive versus Batch Mode

The UTILITY and EXTRACT programs can run either interactively, from a terminal or terminal emulator connected to the HP 3000, or as unattended batch jobs.

The command syntax is the same in either mode: a command followed by one or more parameters. Parameters can be entered in any order; if a parameter has a value associated with it, the value must be entered immediately after the corresponding parameter.

Within Syntax Statements

There are two types of parameters—*required* parameters (for which there are no defaults) and *optional* parameters (for which defaults are provided). How Performance Collection Software handles these parameters depends on the mode in which it is running.

Parameters in Interactive Mode

If an *optional* parameter is not provided in interactive mode, the program will display the default parameter and let you either confirm it or override it.

If a *required* parameter is not provided in interactive mode, the program will prompt you to enter the parameter.

Parameters in Batch Mode

If an *optional* parameter is not provided in batch mode, the program will use the Performance Collection Software default.

If a *required* parameter is not provided in batch mode, the program will terminate. For lists of required program parameters, refer to chapters 3–5.

Example

The following example shows the differences between how the UTILITY program's RESIZE command works in batch mode and in interactive mode.

The RESIZE command lets you set the following parameters:

- Type of log file to be resized.
- Size of new file.
- Amount of empty space to be left in file.
- An action.

The following command resizes the global log file so that it will contain a maximum of 120 days of data with empty space equal to 45 days:

```
RESIZE GLOBAL DAYS=120 EMPTY=45 YES
```

The results will be the same whether you enter this command interactively or from a batch job.

The first parameter—GLOBAL—indicates the type of log file data to be resized. If you do not supply this parameter, the consequent action for interactive and batch users would be the following:

- Batch users.

  The batch job would terminate because this parameter has no default.

- Interactive users.

  You would be prompted to choose which type of log file data to resize to complete the command.

The last parameter—YES—indicates that resizing will be performed unconditionally. If you do not supply this parameter, the consequent action for interactive and batch users would be the following:

- Batch users.

  Resizing would continue as before since YES is the default action.

- Interactive users.

  You would be prompted to supply the action before resizing takes place.

Chapters 4 and 5 contain information about commands and parameters and their defaults. You can also use the online HELP command to check command syntax.

Softkeys and Native Language Support

This section explains two additional features of the Performance Collection Software EXTRACT and UTILITY programs:

- Softkeys.
- Native language support.

Softkeys

Most Hewlett-Packard terminals and terminal emulation programs support user-labeled softkeys. When you run EXTRACT or UTILITY, you can use these softkeys as shortcuts for entering frequently used commands.

Generally, softkeys are loaded any time you run EXTRACT and UTILITY from a Hewlett-Packard terminal. You might want to control how these keys are loaded to adapt them to a particular terminal or emulator program, or you might want to speed up the loading of EXTRACT or UTILITY. Softkeys are not loaded for batch jobs.

The EXTRACT and UTILITY softkeys are controlled by entering a user job control word (JCW). Settings are not saved from one session to another and must be reentered each time you log on.

The following commands take effect the next time EXTRACT and UTILITY are executed from the same session and every time thereafter until another such command is issued or you terminate your session:

`:SETJCW RXKEY=1` *Load softkeys, which override any current information.*

This option allows you to use the softkeys but saves you a few seconds by not saving and restoring the original softkey information. You might use this option if you normally do not use the softkeys except in applications that load them.

`:SETJCW RXKEY=2` *Save current softkeys, use them, then restore originals.*

This is the default setting. Current softkey information is saved, then EXTRACT and UTILITY load their softkeys as they run. When the program terminates, the original softkey information is restored.

This is the most useful option, but it is also the most time consuming. Saving and restoring the original softkey information, however, takes only a few seconds and should not concern most users.

`:SETJCW RXKEY=3` *Do not load or alter softkeys.*

Use this option with UTILITY or EXTRACT on non-Hewlett-Packard terminals or emulators that do not support the softkey feature.

If you do not set this option, each time UTILITY and EXTRACT are executed they will wait about 5 seconds for a response from the terminal before trying to load the softkeys. They may also send an escape code sequence that can cause some terminals to hang.

If you encounter any problems with the softkey loading feature of UTILITY or EXTRACT, disable it with :SETJCW RXKEY=3.

Native Language Support

EXTRACT and UTILITY can support the date and time formats of other native languages if you use the native language support (NLS) feature of the MPE operating system. Consult the *Native Language Programmer's Guide* for additional details.

If a native language is installed and selected on your system, EXTRACT and UTILITY will make the following adjustments:

- Dates and times are entered and printed in the language specified in the NLS package. If a date or time is entered in an unrecognizable format, the program prompts with an example in the correct format.
- Many, but not all, prompts and messages are obtained from an alternate message catalog for the chosen language. If this message catalog cannot be found, the default, LSRXC000, is used.
- The text for the HELP command is obtained from an alternate help catalog for the chosen language. If this help catalog cannot be found, the defaults, EXTRH000 and UTILH000, are used.

Performance Collection Software Host Programs

The Performance Collection Software host programs are SCOPE(XL), SCOPE(XL)2, UTILITY, and EXTRACT. An overview of each is presented below. For detailed command descriptions, refer to chapters 3 (SCOPE), 4 (UTILITY), and 5 (EXTRACT).

Introduction to SCOPE(XL)

The Performance Collection Software SCOPE(XL) program collects and summarizes performance-measurement data on system-resource utilization, terminal transaction rates, and terminal response times. It collects data continuously to provide a complete record of system performance. SCOPE(XL) collects this data from several sources in the HP 3000 system, but it collects data mainly from the HP 3000 Measurement Interface (MI).

The PARM File

The Performance Collection Software PARM file is a standard flat, unnumbered file used to customize your collection environment. The file contains directives (instructions) that tell SCOPE(XL) to log specific performance measurements.

When SCOPE(XL) starts, it looks for the PARM file in the logon group. To enable you to start data collection immediately, Performance Collection Software includes a default PARM file. We suggest that you use this file until you gain experience with Performance Collection Software.

Performance Collection Software Log Files

SCOPE(XL) logs data into one to five log files, depending on the types of information you want to collect:

- Global

  Summarized measurements of the system-wide workload are logged to the file LOGGLOB. One record is written to the log file every 5 minutes.

- Application

  Summarized measurements of the processes in each user-defined group (application) are logged to the file LOGAPPL once every 5 minutes. Users can define up to 15 applications on MPE V or 31 applications on MPE/iX. Performance Collection Software reserves an application called OTHER for recording all processes not included in other user-defined applications.

- Process

  Summarized measurements of selected interesting processes are logged to the file LOGPROC once every minute. A process can become interesting (and be logged) when any of the following occurs:

  - The process is first created.
  - The process terminates.
  - An existing process exceeds user-defined thresholds for CPU, disc, response time, or transaction rate.
  - An existing process exceeds user-defined thresholds for waiting on CPU, disc, or memory, or it is impeded.

- Disc space

  Summarized measurements of disc space usage are logged to the LOGDISC file once a day. This data is an instantaneous “snapshot” of the condition of the system’s disc space at the time the disc was sampled and the peak value for transient or virtual disc space for the day.

  You can control the time of sampling using PARM-file parameters. Disc space is measured for all nonprivate disc volumes (MPE V) and for any private disc volumes (MPE/iX) that are mounted when daily data collection occurs.

Within Syntax Statements

- Indexing information

  Information used to rapidly access the log file data when retrieving information is maintained in the LOGINDX file on MPE/iX systems. On MPE V systems this information is kept in each of the other log files so the LOGINDX file is not needed.

Caution Never modify the LOGINDX file!

You can control how much process history you keep on disc by adjusting the size of each log file with the UTILITY program (see chapter 4). Data collection does not stop when a log file is full. Instead, the oldest 25 percent of the records in the file are deleted to make room for new records.

Note Use only the Performance Collection Software UTILITY program to resize Performance Collection Software log files.

Starting and Stopping SCOPE(XL)

The SCOPE(XL) program resides in the SCOPE.SYS group on the HP 3000. The file SCOPEJOB.SCOPE.SYS is a job stream that runs SCOPE(XL) as the user MANAGER.SYS. If you use the SCOPEJOB file supplied, you must provide all necessary passwords. If you choose to run SCOPE(XL) in any other way, you must set user capabilities to at least the following:

```
   IA, BA, ND, SF, SM
```

When to Start SCOPE(XL)

As mentioned in chapter 1, you should start SCOPE(XL) as soon as possible on the system (or systems) to be measured. This will allow you to develop a background of collected data for assessing performance.

When to Stop SCOPE(XL)

SCOPE(XL) should run continuously. Stop it *only* under the following conditions:

- You are shutting down your system.
- You are updating the Performance Collection Software host software to a new release.
- You are changing the size of a Performance Collection Software log file.
- You are performing a system backup and want to include the Performance Collection Software log files.

  You can perform system backups with SCOPE(XL) running, but Performance Collection Software log files will not be included in the backup. (They are always "busy".) If you RELOAD your system, you lose the contents of these log files unless they have been backed up. See chapter 6 for different archiving strategies you can use to save Performance Collection Software log files.

Data collection also stops if a system failure occurs or SCOPE(XL) aborts. If this occurs, you must follow the restarting procedures described below.

Note At the end of each interval, SCOPE(XL) flushes its file buffers to disc and updates the MPE file label on its log files. Even if it aborts abnormally, you will not lose more than the data collected during the current interval (5 minutes for global and application data and 1 minute for process data).

Restarting SCOPE(XL)

You have three options for restarting SCOPE(XL) after the system has been down or the SCOPEJOB job is canceled:

- You can include the streaming of the file SCOPEJOB as part of SYSSTART.PUB.SYS. It can be streamed each time the system comes up.
- You can build a logon UDC for OPERATOR.SYS that includes the streaming of SCOPEJOB.

Within Syntax Statements

- You can stream SCOPEJOB whenever you want to initiate SCOPE(XL) by including it in your own startup procedure that is executed each time the system comes up.

If you restart SCOPE(XL), Performance Collection Software continues to use the same log files and appends new records to the end of the file.

If you restart SCOPE(XL) while it is already running, the second SCOPE(XL) job will terminate immediately.

Introduction to UTILITY

The UTILITY program serves as a tool for managing and reporting information on log files. The program also lets you display the versions of Performance Collection Software host programs and libraries being used, and control the SCOPE(XL) collection program as it runs.

UTILITY Commands

UTILITY can be used in either an interactive or a batch mode. Brief descriptions of the major functions are given below. For detailed explanations of each command, see chapter 4.

`ALARMS`	Enables or disables processing of performance alarms during the SCAN operation. Performance alarm definitions are located in the last PARM file processed with the PARMFILE command.
`CONVERT`	Converts raw or extracted log files created under previous versions of Performance Collection Software (A.*nn.nn*) to the current format. These files must reside on the host system. Only *forward* conversion is provided. That is, you can only convert log files from an earlier release to this new release. Log files converted to the new release or log files created under this new release cannot be converted to or accessed by earlier releases of Performance Collection Software or HP LaserRX/MPE.
`CREATE`	Creates the four Performance Collection Software raw log files—LOGGLOB, LOGAPPL, LOGPROC, and LOGDISC—before starting SCOPE(XL) for the first time.

Within Syntax Statements

`PARMFILE`	Checks the syntax of a Performance Collection Software PARM file and reports any warnings or errors.
`RESIZE`	Resizes Performance Collection Software raw log files (LOGGLOB, LOGAPPL, LOGPROC, and LOGDISC). The RESIZE command requires that a log file be opened. UTILITY is the *only* program you can use to resize raw log files. Any attempt to use a program other than the UTILITY program to manipulate the size of a log file will destroy the integrity of the log file.
`SCAN`	Scans a raw or extracted log file and prints a summary report on its contents. The SCAN command requires that a log file be opened.
`SCOPE`	Lets you communicate with SCOPE(XL) while it is running.
`VERSIONS`	Displays version information for all the Performance Collection Software host programs and libraries.

Introduction to EXTRACT

The EXTRACT program is a key part of Performance Collection Software data analysis and archiving procedures. The program creates an extracted file that can be either kept on the host system as a remote file or transferred to a PC for local analysis. It can be used in either an interactive or a batch mode.

You can extract data without interrupting data collection. To ensure that you do not inadvertently mix data from different systems, EXTRACT stores all data types in the same extracted file.

EXTRACT lets you do the following:

- Extract data from raw log files to create an **extracted log file**.
- Extract data from previously extracted log files.
- Summarize extracted data.
- Export data from raw and extracted log files for use outside the Performance Collection Software product.

Extracting Data

You can select a combination of data to be extracted, by types—global, application, process, disc—and by starting and ending date and time. Or, you can use one of several commands that automatically determine the selection and summarization criteria for you.

Summarizing Data

You have several options for summarizing extracted data:

- You can summarize detailed raw data and hourly averages.

  Extracted log files that contain hourly averages take less disc space and can be accessed faster than those containing detailed data. You do, however, lose the time granularity associated with detailed data.

- You can extract detailed and summarized data into a single log file.

  Extracted log files that contain both types of data offer full time granularity. You gain speed of access, but you do not save disc space.

Appending Data to an Existing File

You can use EXTRACT to append data to the end of an existing *extracted* log file, but you will be forced to continue the same data types, subsetting, and summarization used in the initial extraction.

The ability to append data to an extracted log file is valuable for archiving log files. You can use extracted log files for archiving Performance Collection Software data in an easily accessible form or for transporting data between systems. Tracking long-term trends requires keeping data for years. See chapter 6 for recommendations on using EXTRACT to archive data.

EXTRACT Commands

Major EXTRACT functions are described briefly below; for a detailed explanation of each command, see chapter 5.

`EXPORT`	Copies selected data from raw or extracted log files into a format that can be recorded and processed by other users.
`EXTRACT`	Copies selected data from raw or extracted log files into an extracted file format. Extracted files can be processed using other LaserRX/MPE programs on other HP 3000 or PC Analysis workstations.
`MONTHLY`	Performs an extraction based on a calendar month. Start date, stop date, and output name are set automatically.
`SHIFT`	Sets daily shift information for an EXTRACT and EXPORT function. It allows you to limit data extraction to certain hours of the day and to exclude weekends.
`START`	Sets the starting date and time for an EXTRACT and EXPORT function.
`STOP`	Sets the stopping date and time for an EXTRACT and EXPORT function.
`WEEKLY`	Performs an extraction based on a calendar week (defined as seven days starting on Monday and ending on Sunday). Start and stop dates and output name are set automatically.
`YEARLY`	Performs an extraction based on a calendar year. Start date, stop date, and output name are set automatically.

3

Data Collection: SCOPE

This chapter details the SCOPE(XL) PARM file directives. It includes a summary of the directives, information on how to change a directive, directions on how to start and stop SCOPE(XL), and an alphabetized reference section.

Note Performance Collection Software directives and parameters can be entered through any combination of uppercase and lowercase letters. Only the first three characters are required. For example, the APPLICATION directive can be abbreviated as APP.

The file PARM.SCOPE.SYS represents a typical sample of a PARM file.

Syntax Summary

The following table summarizes PARM file directive syntax.

Table 3-1. Parameters of Collection Directives

Directive	**Parameter**
ID	*system id*
LOG	GLOBAL APPLICATION PROCESS DISCSPACE

THRESHOLD	CPU=*percent*
	DISC=*rate*
	RESPONSE=*seconds*
	FIRST=*seconds*
	TRANSACTIONS=*count*
	NONEW
	NOKILLED
	NOSHORT [=*seconds*]
	MINTHINK=*seconds*
	MAXTHINK=*seconds*

WAIT THRESHOLD	CPU=*percent*
	DISC=*percent*
	MEMORY=*percent*
	IMPEDE=*percent*

DAILYTIME	*hour* [*:minute*]
MAINTTIME	*hour* [*:minute*]
APPLICATION	*application name*
FILE=	*filename*[,*filename*, ...]
INTERACTIVE=	*filename*[,*filename*, ...]
BATCH=	*filename*[,*filename*, ...]
USER=	*logon*[;*logon*; ...]
QUEUE=	*execution priority* (L,A,B,C,D,E) [, ...]
OR	

Table 3-1. Parameters of Collection Directives (continued)

Directive	Parameter

SERVICE LEVEL
SLA
	FIRST=*seconds*
	RESPONSE=*seconds*

ALARM	*alarmid*
	TYPE=*typeid*
	SEVERITY=*severity number*

IF	*itemid* (<,>,<=,>=) *itemid* FOR *duration* [MINUTES]
THEN	*alarm action*
FINISH	*alarm action*
REPEAT	*minimum repeat minutes*
VAR	*itemid* (+,−,/, ∗) *itemid*
DISCGROUP	*discgroup name*
GROUP=	*groupname*[,*groupname*, ...]

Within Syntax Statements

If the PARM file is not found in the logon group of the user who is running SCOPE(XL) or a parameter is not specified, the default parameters for each directive are used.

The directives and their default parameter values are shown in Table 3-2.

Table 3-2. Default Parameter Values for Collection Directives

Directive	**Default Parameter Value**
ID	Blanks, the HPTrend system handle, or, if present on an MPE/iX system, the HPSYSNAME system variable.
LOG	GLOBAL PROCESS

THRESHOLD	CPU=10.0
	DISC=10.0
	RESPONSE=5.0
	FIRST=1.0
	TRANSACTIONS=100
	(not NONEW) (not NOKILLED)
	(not NOSHORT)
	MINTHINK=0.2
	MAXTHINK=3600

WAIT THRESHOLD	CPU=50
	DISC=50
	MEMORY=25
	IMPEDE=25

DAILYTIME MAINTTIME	23:30 (11:30 pm) None (scheduled maintenance is not performed).
APPLICATION	No applications specified.

SERVICE LEVEL	FIRST= (same as THRESHOLD FIRST)
	RESPONSE= (same as THRESHOLD RESPONSE)
	Defaults apply only if *neither* service level is given.

ALARM	No performance alarms defined.
DISCGROUP	If no disc group is specified, the 20 largest MPE ACCOUNTS (by disc space) become disc groups.

Modifying Directives

You can use a text editor to create or modify PARM.SCOPE.SYS. The following rules and conventions apply:

- Specify a directive *only* if you want to override a default parameter.
- Begin each line in the PARM file with a PARM file directive. PARM file values cannot be entered by themselves.
- Enter the directives into the PARM file in any order. Order is not important, with the following exceptions:
  - If a directive is entered more than once, the last one entered is used.
  - The FILE, INTERACTIVE, BATCH, USER, QUEUE, and SERVICE LEVELS directives must *follow* the APPLICATION directives that they define.
  - The IF, THEN, FINISH, and REPEAT directives must *follow* the ALARM directive they define.
- Use uppercase *or* lowercase letters *or* both for all directives and parameters.
- Use blanks or any nonalphanumeric characters (such as semicolons, commas, and equal signs) to separate parameters in each statement.

Note Since commas are allowed within the parameter string in the USER parameter, they cannot be used to separate different logon strings. You must use another character (such as a space) to separate parameters or use a separate USER line for each logon.

- Comment the PARM file if you like. Blank lines and any lines beginning with an asterisk (*) will be ignored.

To change configuration directives, do the following:

- Create or modify the PARM file in the SCOPE.SYS group on the HP 3000.
- Stop and restart SCOPE(XL) or use the UTILITY program to issue the SCOPE NEWPARM command.

Starting SCOPE(XL)

Start SCOPE(XL) by streaming the job SCOPEJOB.SCOPE.SYS. Do this by typing the following:

```
STREAM SCOPEJOB.SCOPE.SYS
```

Stopping SCOPE(XL)

There are three ways to stop SCOPE(XL) from collecting data:

- Log on as MANAGER.SYS and run the UTILITY program. Issue the following command:

  ```
  SCOPE KILL
  ```

  You will be asked to confirm your request to stop SCOPE(XL). If you confirm, UTILITY issues a programmatic command to stop data collection. It can take as long as 60 seconds to end the collection process. UTILITY notifies you when this happens.

  This is the method of choice and results in the normal termination of data collection.

- Purge the file RUN.SCOPE.SYS.

  Each time SCOPE(XL) is executed, it creates a file called RUN in its logon group. Once a minute, Performance Collection Software checks to see if this file is still present. If it is absent, Performance Collection Software terminates normally. Normal termination means that whatever remains in the collection job stream will then execute.

- Use the ABORTJOB command to abort the SCOPEJOB job stream.

  The rest of the collection job stream is not executed. This is *not* the preferred method for terminating collection.

Reminder SCOPE(XL) should run continuously. You should only stop it if you are preparing to shut down your system, updating the Performance Collection Software host software to a new release, changing the size of a Performance Collection Software log file, or performing a system backup and want to include the Performance Collection Software log files.

Hint The easiest way to stop the SCOPE(XL) collector as a part of another batch job (for example as part of the full system backup job) is to issue the SCOPE KILL command in UTILITY. This command pauses until the collector terminates insuring successful shutdown before the job stream continues. The following command will stop the SCOPE(XL) collector if executed by a user with SM or OP capability:

```
:RUN UTILITY.SCOPE.SYS;INFO="SCOPE KILL"
```

When finished, the batch job can then restart the SCOPE(XL) collector by streaming SCOPEJOB:

```
:STREAM SCOPEJOB.SCOPE.SYS
```

ALARM

$$\texttt{ALARM}\ \textit{alarmid}\ \begin{bmatrix}\texttt{TYPE=}\textit{typeid}\\ \texttt{SEVERITY=}\textit{severity number}\end{bmatrix}$$

The following directives and parameters are discussed in chapter 7:

alarmid
typeid
severity number
`IF` *itemid condition itemid* `FOR` *duration* `[MINUTES]`
`THEN` *alarm action*
`FINISH` *alarm action*
`REPEAT` *minimum repeat minutes*
`VAR` *itemid operator itemid*

APPLICATION

`APPLICATION` $\{$ *application name* $\}$

`FILE=`[*filename*]

`INTERACTIVE=`[*filename*]

`BATCH=`[*filename*]

`USER=`[*job/session name,*] $\{$ *username.acctname* $\}$ [*,groupname*]

`QUEUE` [*execution priority*]

`OR`

`SERVICE LEVEL` $\left[\begin{array}{l}\texttt{FIRST=}\textit{seconds}\\ \texttt{RESPONSE=}\textit{seconds}\end{array}\right]$

where:

The *application name* parameter identifies an application that groups multiple program files together and reports on their combined activities. The *application name* is a string of up to 20 characters identifying that application.

One or more *application definitions* should follow the APPLICATION command. An application definition consists of combinations of FILE, INTERACTIVE, BATCH, USER, or QUEUE parameters.

The OR parameter is used to separate multiple definitions for the same application.

All application definitions apply to the last APPLICATION command entered.

Within Syntax Statements

The parameters for application definitions fall into three categories:

File name:	`FILE`
	`INTERACTIVE`
	`BATCH`

User logon:	`USER`
Execution priority:	`QUEUE`

If a parameter from a category is used in an application definition, a process will not belong to that application unless a match for that category is made.

If no parameters in a category are used in an application definition, then a process does not have to match that category.

For example, there can be any number of FILE and USER parameters following an APPLICATION directive, but if there is no QUEUE parameter, a process could belong to any queue and still belong to the application. It *would* have to match at least one of the FILE parameters *and* one of the USER parameters.

Or, as another example, if an application consisted of USER and QUEUE parameters but had no FILE, INTERACTIVE, or BATCH parameters, then any program file name could belong to the application as long as the user logon *and* execution queue matched one of the USER and QUEUE specifications.

Use the OR parameter when you want more than one application definition to apply to the same application.

You can define up to 15 applications on MPE V and 31 applications on MPE/iX. Performance Collection Software predefines an application called OTHER that collects all processes not defined by APPLICATION directives in the PARM file.

If a process file is included in more than one application, it is logged in the first application in which it is defined.

```
APPLICATION=Order Processing
FILE=@.@.OFICPROD,@.@.PAPRPROD,@.@.SUPPLIES

APPLICATION=Office Products
FILE=@.FURNITUR.OFICPROD,@.ELECTRON.OFICPROD
```

Since @.@.OFICPROD is included in Order Processing, any programs in @.FURNITUR.OFICPROD or @.ELECTRON.OFICPROD will be logged in the Order Processing application, not in Office Products. However, if the order of the applications is reversed, any programs in the groups FURNITUR and ELECTRON in the OFICPROD account will be logged in the Office Products application, not in Order Processing.

By default, no user applications are defined.

Note Since any process on the system can belong to only one application, no process will be counted in more than one application. The PARM file is processed in the order entered, and the first match of a program name, user logon, and queue will define the application to which a particular process belongs.

FILE=*filename*

The FILE parameter specifies which program files belong to an application. It includes all interactive or batch executions of these programs and applies to the last APPLICATION directive issued. An error is generated if no APPLICATION directive is found.

The *filename* can be any of the following:

- A simple, fully qualified MPE file name. For example,

  ```
  EDITOR.PUB.SYS
  ```

- A generic file name. For example,

  ```
  VOODOO#.@.@
  ```

Within Syntax Statements

- A partially-qualified file name. For example,

  ```
  SPOOK5
  ```

  In this case, all groups and all accounts are assumed (SPOOK5.@.@).

Multiple file names can be entered on the same line, separated by commas, or in separate FILE parameters.

INTERACTIVE=*filename*

The INTERACTIVE parameter acts the same as the FILE parameter except it signifies that only interactive executions of the program (sessions) are included in the application class.

BATCH=*filename*

The BATCH parameter acts the same as the FILE parameter except it signifies that only batch executions of a program (jobs) are included in the application class.

USER=[*job/session name,*]{*username.acctname*}[*,group*]

The USER parameter identifies the job or session, user, account, and group to be included in the application class.

- *job/session name,*

  The job/session name is an optional part of a user logon that can be used for identification. If this parameter is included, it must be terminated by a comma. If this parameter is not supplied, all job/session names are matched.

- *username*

  The username specifies the name of the user as defined by the MPE accounting structure.

- *acctname*

  The accountname specifies the name of the logon account as defined by the MPE accounting structure.

- *group*

  The groupname specifies the logon group of a particular job or session. It is optional, but if it is included, it must be preceded immediately by a comma. If this parameter is not supplied, all logon group names are matched.

Note Since commas are allowed within the parameter string in the USER parameter, they cannot be used to separate one logon string from another. You must use another character (such as a space) to separate parameters or use a separate USER line for each logon.

QUEUE [L,A,B,C,D,E] ...

Usually, the execution priority of a process is CS, DS, or ES. Special accounts can log on or run programs with ;PRI=BS, whereas privileged mode and system processes can place themselves in the AS or linear queues. (Linear queue can be any priority you want, but it is not subject to the normal priority adjustments of the MPE dispatcher.)

You can restrict processes in an application to those belonging to selected execution priority queues using the QUEUE parameter. More than one queue can be indicated. The following example specifies any process running in either the DS or ES queue:

```
QUEUE=D,E
```

Note The AS and BS queues are special cases of the linear execution queue. If you specify QUEUE=L, processes executing as AS or BS priority are included. Specifying QUEUE=A,B is not the same as specifying QUEUE=L since the linear queue can be outside the range of the A and B queues.

A process with process-handling capabilities can change its execution queue as it runs. A process's execution priority can be altered also by an external tool, such as HP GlancePlus.

The process execution queue is sampled at the end of each 1-minute sample interval. If the process changed queues, it can change applications. All activity for a process during the 1-minute sample interval is assumed to have occurred in the new queue and as such is attributed to the application that matches the process at the *end* of each 1-minute sample interval.

OR

Use the OR parameter to allow more than one application definition to apply to the same application. Within a single application definition, a process must match at least one of each category of parameters. Parameters separated by the OR parameter are treated as independent definitions. If a process matches the conditions for any definition, it will belong to the application, as in the following example:

```
APPLICATION=CONSOLE
USER=OPERATOR.SYS
OR
USER=MANAGER.SYS
FILE=STORE.PUB.SYS,SYSDUMP.PUB.SYS
```

This defines an application (`CONSOLE`) that consists of any programs run by the user `OPERATOR.SYS` plus two programs (`STORE.PUB.SYS` and `SYSDUMP.PUB.SYS`), *if* they are executed by the user `MANAGER.SYS`.

SERVICE LEVEL [FIRST=*seconds*][RESPONSE=*seconds*]

The SERVICE LEVEL parameter defines the response time or times used in measuring an application's service agreements. See appendix B for a discussion of service levels.

Sample Application Definition

The following is an example of how an application can be defined:

```
APPLICATION=Program Development
INTERACTIVE=EDITOR.PUB.SYS, QEDIT.@.@, VOODOO#, TDP
USER=@.TEST @.@,TESTGROUP LABTEST,@.@
SERVICE LEVEL FIRST=1.0

APPLICATION=Other Editors
FILE=EDITOR.PUB.SYS, QEDIT.@.@, VOODOO#, TDP
APPLICATION=Compilers
FILE=COBOL@.PUB.SYS,FORTRAN.PUB.SYS,FTN@.PUB.SYS
FILE=PASCAL.PUB.SYS, SPL.PUB.SYS, RPG.PUB.SYS

APPLICATION=HP Products
FILE=@.PUB.SYS
```

The following is an example of how programs would be logged using the preceding PARM file:

Program	User Logon	Application
EDITOR.PUB.SYS	GERRY.TEST	Program Development
EDITOR.PUB.SYS	SAM.SNEED	Other Editors
HPSLATE.PUB.SYS	SAM.SNEED	HP Products
PASCAL.PUB.SYS	GERRY.TEST	Compilers
EDITOR.UTIL.SYS	GERRY.TEST	OTHER

If the HP products' application in the previous example was defined first, the first four examples would belong to it.

DAILYTIME

`DAILYTIME` *hour* [*:minute*]

where *hour* is a one- or two-digit number from 1 to 24, and *minute* is a two-digit number from 00 to 59. Specify the time in a 24-hour-clock format: midnight=00:00 and noon=12:00.

The default time for daily collections is

```
DAILYTIME 23:30
```

(The time, 23:30, is 11:30 PM.)

The DAILYTIME directive sets the time for the daily data collection. Certain types of data, such as disc space, are only sampled and logged once a day. On some systems such as MPE V, this sampling can use significant amounts of system resources and should be scheduled for times when it is likely that the system will be idle.

You can specify the time for the daily data collection in two formats:

- DAILYTIME 22 specifies data collection should occur at 22:00 hours (10:00 PM).

  Use this format to specify times on the hour.
- DAILYTIME 22:12 specifies data collection should occur 12 minutes after 22:00 hours or 10:12 PM.

  Use this format to specify times other than on the hour. Simply add a colon and the minutes.

DISCGROUP

DISCGROUP=$\{$ *discgroup name* $\}$

The following related parameter is also discussed in this section:

GROUP=$\{$ *groupname* $\}$

The DISCGROUP directive defines a set of MPE ACCOUNTS and GROUPS that will have their disc space usage reported as a unit under the assigned disc group name.

The disc group name can be up to 8 characters long and can contain uppercase and lowercase alphanumeric characters.

A maximum of 20 disc groups can be defined. There is no automatic OTHER disc group to contain all unaccounted disc space. If you want to account for all disc space on a system, define the last disc group as follows:

```
DISCGROUP=OTHER
GROUP=@.@
```

If you do not use DISCGROUP, the 20 largest MPE ACCOUNTS (by disc space) will be grouped by default.

GROUP

`GROUP=`$\{$ *groupname* $[$`,`*groupname*`,` ... $]\}$

The GROUP parameter lets you specify the MPE ACCOUNTS and GROUPS that are to be included in the set.

You can only use GROUP if it is preceded by the DISCGROUP directive.

The following are valid *groupnames*:

- Simple, fully-qualified MPE GROUP names, such as

  ```
  PUB.SYS
  ```

- Generic GROUP names, such as

  ```
  WORK#.@
  ```

- Partially-qualified GROUP names, such as:

  ```
  PUB
  ```

  In this case, the PUB GROUP in all accounts is assumed (PUB.@).

(For a discussion of generic filenames and the use of wild cards see the *MPE/iX Commands Reference Manual*.)

ID

`ID=`[*system id*]

where *system id* is a string of up to 40 characters that identifies your system. If you have multiple systems, use different ID strings for each one. The first 10 characters are displayed on Performance Collection Software graphs.

The default for ID is all spaces, with the following exceptions:

- If HPTrend is configured on the system, the default is the HPTrend system handle.
- On MPE/iX systems, if the HPSYSNAME system variable is present, it is used as the default ID.

Note If you did not assign a *system id* when you defined the ID parameter, SCOPE(XL) will issue a warning message when it executes.

LOG

```
LOG [GLOBAL][APPLICATION][PROCESS][DISCSPACE]
```

The default is `LOG GLOBAL PROCESS`.

GLOBAL

If GLOBAL is specified, global records are written to the LOGGLOB log file.

APPLICATION

If APPLICATION is specified, application records are written to the LOGAPPL log file.

PROCESS

If PROCESS is specified, interesting processes are written to the LOGPROC log file. A process can become interesting when it is first created, when it terminates, or when it exceeds certain user-defined thresholds.

DISCSPACE

If DISCSPACE is specified, a daily measurement of the system's disc space is written to the LOGDISC log file. The DAILYTIME directive in the PARM file controls when this measurement is taken.

The log files are created automatically if logging to them is specified and they do not already exist. If a particular type of logging is disabled, the corresponding log file is not purged.

Note The default for logging performance data is the indication that no applications are specified. To be able to graph application data, you must define your particular application subsets.

To log disc space information, the disc space collector program must be located in the same group as SCOPE(XL). On MPE V systems the program is called SCOPE2. On MPE/iX systems the program is called SCOPEXL2. These programs are furnished with the Performance Collection Software update files.

MAINTTIME

```
MAINTTIME hour [:minute]
```

where *hour* is a one- or two-digit number from 1 to 24, and *minute* is a two-digit number from 00 to 59. Specify the time in a 24-hour-clock format with midnight=00:00 and noon=12:00.

As the default, daily scheduled maintenance *will not* be performed.

The MAINTTIME directive sets the time for daily maintenance procedures. If a valid time is specified, the following activities will be performed by the SCOPE(XL) collector daily at that time:

- Global, Application, and Process log files are examined to estimate if they might be filled within the next 24 hours. If so, a log file roll back is performed immediately rather than waiting until the next day. This ensures that log file roll back operations occur at scheduled times rather than at random.
- Scratch disc files that remain after remote log file access are purged. On MPE/iX, these files are named uniquely as HCLS followed by four numbers, and on MPE V, as HCLST followed by three numbers. Only files in the logon group for the SCOPE(XL) collector are accessed, and only empty files are purged.

THRESHOLD

```
          ⎡CPU=percent          ⎤
          ⎢DISC=rate            ⎥
          ⎢RESPONSE=seconds     ⎥
          ⎢FIRST=seconds        ⎥
THRESHOLD ⎢TRANSACTIONS=count   ⎥
          ⎢NONEW                ⎥
          ⎢NOKILLED             ⎥
          ⎢NOSHORT[=seconds]    ⎥
          ⎢MINTHINK=seconds     ⎥
          ⎣MAXTHINK=seconds     ⎦
```

The parameters for THRESHOLD can be entered on the same line (separated by commas) or on separate THRESHOLD lines.

The default for THRESHOLD is the following:

```
THRESHOLD CPU=10.0, DISC=10.0, RESPONSE=5.0, FIRST=1.0
THRESHOLD TRANSACTIONS=100
THRESHOLD MINTHINK=0.2, MAXTHINK=3600
```

Note The default status for NONEW, NOKILLED, and NOSHORT parameters is OFF (*not enabled*)—that is, NEW, KILLED, and SHORT processes *will* be logged.

CPU=*percent*

This parameter sets the amount of CPU utilization that a process must exceed to become interesting and be logged. It is only used if process logging is enabled.

Percent (1.0–100.0) indicates overall CPU usage. For example, CPU=7.5 indicates a process will be logged if it exceeds 7.5 percent CPU utilization in a 1-minute sample interval.

DISC=*rate*

Rate (1–32767) signifies the disc I/O rate in transfers per second.

The DISC parameter sets the rate of physical disc I/Os that a process must exceed to become interesting and be logged.

For example, DISC=8.0 indicates a process will be logged if it exceeds an average of 8 disc I/Os per second in a 1-minute sample interval.

RESPONSE=*seconds*

Seconds (0.1–32767) signifies terminal response time.

The RESPONSE parameter sets the average terminal response time that a process must exceed to become interesting and be logged. Terminal response time is the number of seconds between pressing (Return) or (Enter) and the next prompt.

This parameter also sets the default value for an application's service-level response time.

For example, RESPONSE=4.0 indicates a process will be logged if its response time exceeds an average of 4 seconds in a 1-minute sample interval.

FIRST=*seconds*

Seconds (0.1–32767) signifies time-to-first response.

The FIRST parameter sets the average time-to-first response that a process must exceed to become interesting and be logged. Time-to-first response is the number of seconds between pressing (Return) or (Enter) and the first data are written to the terminal.

This parameter also sets the default value for an application's service-level response time.

For example, FIRST=1.5 indicates a process will be logged if its first response time exceeds an average of 1.5 seconds in a 1-minute sample interval.

TRANSACTIONS=*count*

Count (0–32767) signifies the number of terminal transactions.

The TRANSACTIONS parameter causes any process that completed more than the specified number of terminal transactions during a 1-minute sample to become interesting and be logged.

For example, TRANSACTIONS=85 indicates a process will be logged if it exceeds 85 transactions in a 1-minute sample interval.

NONEW

The NONEW parameter prevents the logging of any process that would have been considered interesting *only* because it was a new process *and* the process was not interesting for any other reason during the sample interval.

You can use the NONEW parameter to reduce the amount of disc space required to hold process information when processes are being created rapidly.

NOKILLED

The NOKILLED parameter prevents the logging of any process that would have been considered interesting *only* because it was a terminating process *and* the process was not interesting for any other reason during the sample interval.

You can use the NOKILLED parameter to reduce the amount of disc space required to hold process information when processes are terminating frequently.

NOSHORT[=*seconds*]

Seconds (1–32767) signifies the minimum time a process must run to be logged. The default is 30 seconds.

The NOSHORT parameter prevents the logging of any process that would have been considered interesting *only* because it was created or terminated *and* has a run time less than or equal to the specified number of seconds.

The process will be logged normally if it was interesting for any reason other than being new or killed, or if it ran for longer than the specified time.

Use the NOSHORT parameter to reduce the amount of disc space required to hold process information when many short-duration processes are executed. If you use NOSHORT instead of NONEW and NOKILLED, you can eliminate short processes from the log file while still logging the start and stop of longer-running processes.

MINTHINK=*seconds*

The range for the MINTHINK value is 0.001–60.000 seconds.

The MINTHINK parameter sets a filter used in calculating terminal response times. Any terminal transaction having a think time less than the MINTHINK setting is assumed to be a hardware-generated transaction, such as a terminal status read.

Since the user at a terminal does not see or respond to this read to the terminal, the user does not consider it to delimit a real terminal transaction. Terminal reads that complete in less than the minimum think time are therefore not used to delimit terminal transactions, and they are treated the same as if they were a terminal write.

The following factors affect the MINTHINK parameter setting:

- Increasing the number of seconds in the MINTHINK parameter reduces the number of terminal transactions and increases the average response times.
- Setting MINTHINK too high can eliminate actual user-entered terminal transactions and can report response times that are too high.
- Setting MINTHINK too low can cause hardware-satisfied terminal reads—such as terminal status requests used often in block mode protocols like VPLUS—to be counted as terminal transactions. This can result in a higher transaction count than is perceived by the user and a shorter average response time.
- Setting MINTHINK to 0.0 defeats the minimum transaction think-time filtering mechanism, and every terminal read will count as a transaction. You may need to increase MINTHINK to account for delays caused by data communications to terminals.

MAXTHINK=*seconds*

The range for the MAXTHINK value is 0–86400 seconds (24 hours).

The MAXTHINK parameter sets a filter used in calculating terminal response times. Any terminal transaction having a think time greater than the MAXTHINK setting is ignored. All times involved with this transaction (think time, time-to-first response, and response time) are discarded and a new transaction is begun.

You can use the MAXTHINK directive to eliminate start-up transactions at a terminal after a long absence. If a terminal application is left idle for a long time, MPE may reallocate the memory resources of that application to other users on the system. When the next transaction is completed all those memory resources must be re-acquired, often resulting in an abnormally long response time.

To ignore the first transaction following a long absence, you can set the MAXTHINK parameter to the number of seconds you want to consider a long absence.

WAIT THRESHOLD

```
               ┌ CPU=percent    ┐
WAIT THRESHOLD │ DISC=percent   │
               │ MEMORY=percent │
               └ IMPEDE=percent ┘
```

where *percent* indicates the portion of the 1-minute sample interval during which the process was waiting.

The default for WAIT THRESHOLD is the following:

```
WAIT THRESHOLD CPU=50, DISC=50, MEMORY=25, IMPEDE=25
```

The WAIT THRESHOLD directive sets the time a process must wait for a certain resource before becoming interesting and being logged.

Do not confuse the parameters of the WAIT THRESHOLD directive with those of the THRESHOLD directive since many of them are similar. THRESHOLD parameters set the amount of a given resource that a process must *use* in order to be logged. WAIT THRESHOLD parameters set the length of time a process must *wait* for a resource in order to be logged. WAIT THRESHOLD parameter values are expressed as the percentage of time during the 1-minute sample interval that the process waited for the resource. *Percent* values can range between 1.0 and 100.0 percent.

CPU=*percent*

This parameter sets how long a process must wait for access to the CPU before becoming interesting and being logged. It is only used if process logging is enabled.

For example, WAIT CPU=10.5 indicates a process will be logged if it must wait for access to the CPU for more than 10.5 percent of a 1-minute sample interval.

Within Syntax Statements

DISC=*percent*

This parameter sets how long a process must wait for data transfer to or from a disc drive before becoming interesting and being logged.

For example, WAIT DISC=50 indicates a process will be logged if it must wait for disc for more than half (50 percent) of the 1-minute sample interval.

MEMORY=*percent*

This parameter sets how long a process must wait for code or data to be brought into main memory before becoming interesting and being logged.

IMPEDE=*percent*

This parameter sets how long a process must wait in the IMPEDE state during the 1-minute sample interval before becoming interesting and being logged.

The IMPEDE state is a software synchronization mechanism and can be used for many situations, such as synchronizing access to the HP TurboINMAGE databases.

4

Data Management: UTILITY

This chapter gives details on the UTILITY commands. It includes a command syntax summary, a discussion of the SCAN command, and a command reference section. (The reference section lists the commands in alphabetical order.)

Note Commands and parameters for UTILITY can be entered with any combination of uppercase and lowercase letters. Only the first three letters of the command's name are required. For example, the command CONVERT can be entered as `CONVERT` or it can be abbreviated to `CON`.

Examples of various tasks using the UTILITY program can be found in the programs's online HELP facility.

UTILITY Command Syntax Summary

The following table contains a summary of UTILITY commands syntax and their parameters.

Table 4-1. UTILITY Commands: Syntax and Parameters

Command	Parameter
ALARMS	ON OFF
CONVERT	*logfile*
CREATE	GLOBAL APPLICATION PROCESS DISC DAYS *maxdays* SIZE *maxMB*
DETAIL	ON OFF
EXIT E	
HELP	*topic*
LIST	*device*
LOGFILE	*logfile*
MENU	
MPE :	*command*
NOTES	ON OFF *level*

Table 4-1.
UTILITY Commands: Syntax and Parameters (continued)

Command	Parameter
PARMFILE	*parmfile*
RESIZE	GLOBAL APPLICATION PROCESS DISC DAYS *maxdays* SIZE *maxMB* EMPTY *days* SPACE *MB* YES NO MAYBE
SCAN	*logfile* Operation is also affected by the LIST, START, STOP, DETAIL, ALARMS, and NOTES commands.
SCOPE	STATUS KILL NEWPARM NOTE *message*
SHOW	ALL
START	*date* [*time*] TODAY [-*days*][*time*]
STOP	*date [time]* TODAY [-*days*][*time*]
VERSIONS	*group*

ALARMS

$$\texttt{ALARMS} \begin{bmatrix} \texttt{ON} \\ \texttt{OFF} \end{bmatrix}$$

You can set the alarms ON or OFF to select whether or not to print individual performance alarm events in the SCAN report. See the description of the SCAN command for information on this report. See also chapter 7 on Performance Alarms.

The default is ALARMS=ON if alarm definitions were found during the last PARMFILE command.

CONVERT

CONVERT $\{$ *logfile* $\}$

This command converts one or more raw or extracted log files to the current version's format. The MPE log file format was altered starting with the B.00.*nn* release of Performance Collection Software. Earlier versions of the log files are known as *version A* log files, whereas the B.00.00 log files are *version B* log files.

Only *forward* conversion is provided. That is, you can only convert log files of earlier releases to those of this release. Log files converted to the new release or created under this new release *cannot* be converted or accessed by earlier releases of HP LaserRX/MPE or the Performance Collection Software.

You must be logged on to the same group and account, and have exclusive write access to the log files in order to convert them.

Caution Your existing log files will be altered by the conversion process. If you did not back up your log files when you installed this release, you should back them up before you convert them. See chapter 1.

If you try to access a version A log file with a version B program or library you will receive error message 1005 (invalid log file version; see "Introduction to UTILITY" in chapter 2).

To convert a raw or extracted log file, do either of the following:

CONVERT *logfile*

or

LOG *logfile*
CONVERT

You may receive the error message when the file is opened but before conversion. If you try to convert an already-converted file, nothing will happen.

During the log file conversion process, you must have exclusive access to the log files. The SCOPE(XL) data collection program must be stopped before converting *raw* log files, but before converting extracted log files.

Within Syntax Statements

Since the UTILITY program runs on the HP 3000 system, the parent log files of any log file downloaded to the PC should be converted on the HP 3000, then downloaded again. If the parent log file does not still exist on the HP 3000, the log file can be uploaded from the PC *if care is taken* to transfer BINARY data to a file with the proper MPE file characteristics (file code, record size, blocking factor, etc.). Once transferred back to the HP 3000, it can be converted then downloaded again.

The conversion process makes a copy of the file during the conversion. Be sure that sufficient disc space is available on the system before performing a conversion.

Note You will need enough free disc space equal to the size of the file being converted.

Raw log files *always* should be converted on the same type of system (MPE V or MPE/iX) as they were created. Extracted log files can be converted on any type of system.

The conversion process adjusts log file sizes without losing data. The CONVERT command requires an open log file. If a log file is not opened, you will be prompted to open one (in interactive mode) or the batch job will terminate (in batch mode).

Sample CONVERT Task

In this sample task, an extracted log file is converted into a form compatible with the current release of Performance Collection Software.

1. `LOGFILE RXLOG`

   RXLOG is still in the old format. Error 1005 indicates that it is an invalid version. RXLOG must be converted to the new format before it can be opened.

2. `CONVERT`

   RXLOG is converted to the new format.

CREATE

$$\texttt{CREATE}\left\{\begin{array}{l}\texttt{GLOBAL}\\ \texttt{APPLICATION}\\ \texttt{PROCESS}\\ \texttt{DISC}\end{array}\right\}\left[\begin{array}{l}\texttt{DAYS=}\mathit{maxdays}\\ \texttt{SIZE=}\mathit{maxMB}\end{array}\right]$$

The CREATE command produces the four Performance Collection Software raw log files—LOGGLOB, LOGAPPL, LOGPROC, LOGDISC—before starting SCOPE(XL) for the first time. These log files *must not already exist* or any attempt to create them will fail.

You can use CREATE to produce a log file with a specified maximum size. If SCOPE(XL) is initiated with a logging type enabled and a log file does not exist, one is created in the default size. You can use the UTILITY CREATE command to create log files before running SCOPE(XL).

The log type to be created is a required parameter for batch mode execution. If CREATE is being executed interactively, and this parameter is defaulted, you are prompted as to whether you wish to create each type of log file in turn.

The maximum file size can be specified using the DAYS (in days) or SIZE (in megabytes) parameter. If days are specified, a default megabytes-per-day value is used based on the type of the log file being created. If DAYS and SIZE are both entered, SIZE is used.

The default maximum size is 20 megabytes for global, application, and process log files and 1 megabyte for disc log files.

Caution Log files created by UTILITY have data structures specific to the type of system on which UTILITY is run (MPE V versus MPE/iX). You should not create log files on one type of system for use on another type of system.

The LOGINDX file is created on MPE/iX systems whenever the first log file of any type is created. (The LOGINDX file is not used on MPE V.)

Log files created by UTILITY are not initialized. SCOPE(XL) initializes them when it is first run.

Sample CREATE Task

The following example shows how to create the four Performance Collection Software raw log files.

You must log on to the group where the log files will reside.

The maximum size of each file is specified in order to override the defaults used by SCOPE(XL).

```
:HELLO SCOPE.SYS,SCOPE
:RUN UTILITY
CREATE GLOBAL DAYS=120
CREATE APPLICATION SIZE=10
CREATE PROCESS SIZE=35
CREATE DISC DAYS=365
```

Notice that you can specify the maximum size of the raw log file in terms of either days (DAYS) or megabytes (SIZE).

DETAIL

```
       ┌    ┐
       │ON  │
DETAIL │OFF │
       └    ┘
```

You can control the level of detail printed in the SCAN and PARMFILE reports by setting the detail to ON or OFF. See the descriptions of the SCAN and PARMFILE commands for specific actions on these reports.

The default is DETAIL=ON.

EXIT

```
EXIT or E
```

To terminate the UTILITY program, enter the EXIT command, or press F8 if you are using terminal softkeys.

HELP

```
HELP [topic]
```

Access the online HELP facility interactively by entering the HELP command or pressing F7. This facility and the MPE HELP facility work in the same way. You can enter parameters to obtain information on the UTILITY program's commands and tasks, or on HELP itself.

If you are using a terminal with HP softkey support, you can get help on a softkey's functions by pressing the softkey while in the HELP subsystem. Pressing F8 (`Exit HELP`) exits the HELP subsystem and returns you to the UTILITY program, or you can exit the HELP system by typing `EXIT`.

You can also request HELP on a specific topic. For example,

```
HELP TASKS
```

or

```
HELP RESIZE PARMS
```

When you use this form of the HELP command, you will receive the help text for the specified topic and remain in the UTILITY command entry context. Since you do not enter the HELP subsystem interactively, you will not have to exit or stop before entering the next UTILITY command.

LIST

```
LIST [device]
```

You can change the list device for all UTILITY reports in two ways:

- You can issue a file equation before running the UTILITY program.

```
  :FILE RXLIST;DEV=LP
```

or

- You can use the LIST command once UTILITY is running.

```
  LIST LP
```

The device parameter on the LIST command must represent a valid configured device on the current HP 3000 system. It should be a device that can be written to (normally a printer). If you need to specify more than just the device name, you can issue a file equation for file RXLIST, and then enter the device using the LIST command.

```
MPE FILE RXLIST;ENV=LP2.HPENV.SYS
LIST HP2680
```

To return the listing device back to the standard list device, enter the following:

```
LIST TERM
```
or
```
LIST $STDLIST
```
or
```
LIST *
```

To determine the current list device, enter the LIST command without parameters:

```
LIST
```

If the LIST device is not $STDLIST, most commands are echoed as they are entered.

Sample LIST Task

When you print a summary report on the extracted log file RXLOG to the line printer (device LP), the following will occur:

```
LOGFILE RXLOG     (Opens the RXLOG log file.)
LIST LP           (Directs the SCAN report listing to LP.)
DETAIL OFF        (Specifies less than full detail in the report.)
SCAN              (Reads RXLOG and produces the report.)
```

LOGFILE

`LOGFILE` [*logfile*]

For many UTILITY program functions, a log file must be opened. This can be accomplished by explicitly issuing the LOG command or implicitly by issuing some other commands. If a log file name is not provided, the default LOGGLOB.SCOPE.SYS is used.

You can specify the name of either a raw or extracted log file. If you specify an *extracted* log file name, all information is obtained from this single file. If you specify a *raw* log file name, you must specify the name of the global log file, even if you were not logging global data.

The other raw log files are assumed to be in the same group as the global log file. They have the following names:

`LOGGLOB`	Global log file.
`LOGAPPL`	Application log file.
`LOGPROC`	Process log file.
`LOGDISC`	Disc Space log file.
`LOGINDX`	Index file (MPE/iX only).

Once a log file is opened successfully, a report is printed or displayed showing the general content of the log file (or log files), as in the following example:

```
GLOBAL      FILE: LOGGLOB.SCOPE.SYS           VERSION B
APPLICATION FILE: LOGAPPL.SCOPE.SYS
PROCESS     FILE: LOGPROC.SCOPE.SYS
DISC SPACE  FILE: LOGDISC.SCOPE.SYS
INDEX       FILE: LOGINDX.SCOPE.SYS
SYSTEM ID: COOKIE PTC Hewlett-Packard SERIES 950

DATA COLLECTOR: XL B.02.00
DATA COVERS:   129 DAYS FROM 01/03/90 TO 05/11/90
GLOBAL APPLICATION PROCESS DISCSPACE DATA RECORDS ARE AVAILABLE

THE FIRST GLOBAL      RECORD IS ON 01/03/90 AT 12:40 PM
THE FIRST APPLICATION RECORD IS ON 01/03/90 AT 12:40 PM
THE FIRST PROCESS     RECORD IS ON 01/22/90 AT 10:42 AM
THE FIRST DISC SPACE  RECORD IS ON 08/11/89 AT  6:16 PM
```

You can verify the log file that you opened with the SHOW command, as described later.

You can open another log file at any time by entering another LOG or LOGFILE command. Any currently open log file is closed before the new log file is opened.

The following commands require a log file to be opened:

SCAN
RESIZE
CONVERT

If no log file is currently open, the following commands do an implicit LOGFILE command:

SCAN
RESIZE
CONVERT
CREATE *(following the creation of new raw log files)*

Caution Do not *rename* raw log files! Access to these files assumes the standard log file names are in effect. If you must rename log files—for example to place log files from multiple systems on the same HP 3000 for analysis—first *extract* the data, then rename the extracted log files.

You can use a file equation to open an *extracted* log file, but you should not use file equations to access *raw* log files.

Note If you must have more than one set of raw log files on the same HP 3000 system, create a separate group for each set of files. Although the log file names cannot be changed, the group and account do not have to be SCOPE.SYS. If you wish to alter the log files in any way, you must be logged on to the group where the log files reside and have read/write access to all the log files.

MENU

```
MENU
```

The MENU command prints a short list of the available UTILITY commands.

Command	**Function**
`HELP`	Get the information on commands and options.
`LOGFILE`	Specify a log file to be analyzed.
`PARMFILE`	Specify a Performance Collection Software parameters file to be checked.
`START`	Specify a starting date and time for SCAN.
`STOP`	Specify an ending date and time for SCAN.
`DETAIL`	Enable or disable the printing of details for SCAN.
`ALARMS`	Enable or disable the processing of user alarms during SCAN.
`NOTES`	Set the level of notes listed during SCAN.
`SHOW`	Show the current program settings.
`LIST`	Specify the listing device for SCAN.
`SCAN`	Process the log files and report on contents.
`RESIZE`	Resize the log files. (Do a SCAN first.)
`CREATE`	Create new log files in logon group.
`SCOPE`	Interact with the Performance Collection Software data collection program.
`VERSIONS`	Print the version numbers of all HP LaserRX/MPE and Performance Collection Software host programs.
`MPE` *or* `:`	Execute an MPE command.
`MENU`	List the command menu.
`EXIT`	Terminate this program.
`CONVERT`	Convert the log files to the new format.

MPE

`MPE` *or* : [*command*]

You can enter an MPE command without exiting UTILITY by entering `MPE` or a colon (:) followed by a *valid* MPE V or MPE/iX command.

```
MPE SHOWJOB
:TELL MANAGER.SYS; HI
:EDITOR
MPE RUN FCOPY.PUB.SYS
```

Commands that *cannot* be executed in this fashion are as follows:

ABORT	DO	HELP	SETCATALOG
BYE	EOD	JOB	SHOWCATALOG
CHGROUP	EOJ	LISTREDO	REDO
DATA	EXIT	OPTION	RESUME

UDCs, COMMAND FILEs, and implicit RUN commands can be executed on MPE/iX but not on MPE V.

On MPE V and MPE/iX, the explicit RUN command is allowed. If you omit the command, you will be prompted to supply it:

```
MPE
Enter MPE Command: SHOWJOB
```

NOTES

$$\texttt{NOTES} \begin{bmatrix} \texttt{ON} \\ \texttt{OFF} \\ \textit{level} \end{bmatrix}$$

You can add notes to the Performance Collection Software global log file with the UTILITY program's SCOPE NOTE command. The notes added with this command are reported when a SCAN command is executed.

The SCOPE(XL) program can also write notes to this file to record important events. The NOTES command can control which of these notes are listed during a SCAN function:

- If `NOTES=OFF`, notes are *not* printed during a scan.
- If `NOTES=ON`, *all* notes are printed during a scan.
- If `NOTES=`*level*, only notes set at a level that matches the one entered are printed during a scan.

Current notes levels are as follows:

- 100 (*user note, entered through UTILITY*).
- 102 (*SCOPE(XL) is shutting down*).
- 105 (*SCOPE(XL) was instructed to reread its PARM file*).

PARMFILE

`PARMFILE` [*parmfile*]

Use the PARMFILE command to check the syntax of a Performance Collection Software PARM file and report any warnings or errors.

You can use the PARMFILE command to do any of the following:

- Examine a Performance Collection Software PARM file for errors or warnings.
- Discover how much room is left for defining applications.
- Discover how much room is left for defining disc groups.
- Define Performance Alarms to be used during the SCAN function.
- If `DETAIL=ON`, print the effective contents of the PARM file.

In batch mode:

- If no PARM file name is specified, the default is PARM in the same group and account as the currently-opened log file.
- If no log file is open, the default PARM file name is the file named PARM in the logon group.

In interactive mode:

- If no PARM file name is supplied, you are prompted to supply one.

Sample PARMFILE Task

Check the syntax of the current PARM file (PARM.SCOPE.SYS) and report any warning or errors.

```
DETAIL=ON              (Gives full details on the PARM report.)
PARMFILE PARM.SCOPE.SYS
```

RESIZE

$$\texttt{RESIZE}\left\{\begin{array}{l}\texttt{GLOBAL}\\ \texttt{APPLICATION}\\ \texttt{PROCESS}\\ \texttt{DISC}\end{array}\right\}\left[\begin{array}{l}\texttt{DAYS=}\mathit{maxdays}\\ \texttt{SIZE=}\mathit{maxMB}\end{array}\right]\left[\begin{array}{l}\texttt{EMPTY=}\mathit{days}\\ \texttt{SPACE=}\mathit{MB}\end{array}\right]\left[\begin{array}{l}\texttt{YES}\\ \texttt{NO}\\ \texttt{MAYBE}\end{array}\right]$$

Use the RESIZE command to resize raw log files. This is the *only* program you should use to resize the log files in order to preserve coordination between the files and their internal control structures. If you use other tools, such as FCOPY, you might remove or destroy the validity of these control structures.

Caution UTILITY *cannot* be used to RESIZE extracted files. If you want to resize an extracted file, you should use EXTRACT with appropriate file equations to create a new extracted log file.

The RESIZE command requires a log file be opened. In order to resize raw log files, the files cannot be opened by any other process. You must shut down SCOPE(XL) before resizing log files.

Open the raw log files with the LOGFILE command before issuing the RESIZE command. The RESIZE command allows you to specify log file space in units of megabytes or days. The translation of days to megabytes is more accurate if you do a SCAN command before entering the RESIZE command.

The RESIZE function creates the new file before deleting the original file. Make sure there is sufficient disc space on the system to hold the original *and* the resized log file before doing the resizing procedure.

The default resizing parameters are shown in table 4-2.

Table 4-2. Default Resizing Parameters

Parameter	If Executed Interactively	If Executed in Batch
Log file type	You are prompted for each available log file type.	No default. This is a required parameter.
DAYS SIZE	The current file size.	The current file size.
EMPTY SPACE	The current amount of empty space or enough empty space to retain all data currently in the file, whichever is smaller.	The current amount of empty space or enough empty space to retain all data currently in the file, whichever is smaller.
YES NO MAYBE	You are prompted following the reported disc space results.	Yes. Resizing will occur.

Log File Type Parameter

The log file type parameter specifies the type of data you want to resize. The following raw log files correspond to the current data types:

Data Type	**Raw Log File**
`LOGGLOB`	Global Data Type.
`LOGAPPL`	Application Data Type.
`LOGPROC`	Process Data Type.
`LOGDISC`	Disc Data Type.

Within Syntax Statements

If you do not specify a data type, the batch job terminates (in batch mode), or you are prompted to supply the data type based on those log files that currently exist (in interactive mode).

Size Parameters

Specify two sizes when you resize a log file:

- Maximum size of the file (specify DAYS or SIZE).
- Current amount of empty space required (specify EMPTY or SPACE).

DAYS, SIZE, EMPTY, and SPACE parameters can be entered in units of days or megabytes. Days are converted to megabytes by using an average-megabytes-per-day value for the log file. This conversion factor varies depending on the type of data being logged and the particular characteristics of your system.

Usually more accurate average-megabytes-per-day conversion factors can be obtained if you perform a SCAN command on the existing log file before you do the RESIZE command. A SCAN measures the accumulation rates for your system.

If no SCAN is done or if the measured conversion factor seems unreasonable, the RESIZE command will use a default conversion factor for each type of data.

The DAYS and SIZE parameters specify the maximum size of the log file. The actual size depends on the amount of data in the file. MPE usually allocates disc space in units called extents, with a new extent being allocated when the current one is filled. The default log files are built with a maximum of 32 extents so that the minimum size of a log file is one-thirty-second of its maximum size.

The EMPTY and SPACE parameters specify the minimum amount of room required in the file after the resizing operation is complete. This value is used to determine if any of the data currently in the log file must be removed in the resizing process. You might reasonably expect that a log file would not fill up until the specified number of days after a resizing operation. You might want to use this feature of the RESIZE command to minimize the number of times a log file must be resized by SCOPE(XL) since resizing can occur any time the file is filled. Using UTILITY to force a certain amount of empty space in a log file will cause a log file to be resized when you want it to be.

After resizing, a log file consists of data plus empty space. The data retained is calculated as the maximum file size minus the required empty space. Any data removed during the resizing operation is lost. To save log file data for longer periods of time, use EXTRACT to copy this data to an extracted file *before* doing the RESIZE function.

Action Parameter

The last parameter on the RESIZE command specifies the action to be performed.

- `YES` specifies resizing should be unconditionally performed. This parameter is the default action if UTILITY is not run interactively. If no action is specified when UTILITY is running interactively, you are prompted to supply the action after the resizing report is displayed.
- `NO` specifies resizing should not be performed. This parameter can be specified as an action if you want to see the resizing report but do not want to perform the resizing at this time.
- `MAYBE` specifies that UTILITY should decide whether or not to resize the file. This parameter forces UTILITY to make this decision based on the current amount of empty space in the log file (before any resizing) and the amount of empty space specified in the RESIZE command. If the current log file contains at least as much empty space as specified, resizing is not carried out. If the current log file contains less than the specified empty space, resizing occurs.

  The MAYBE action is intended primarily for use by periodic batch jobs. For example, if you want to ensure that the log files do not fill up during the upcoming week (forcing SCOPE(XL) to resize them during prime-time processing), you could run a batch job that specified a minimum amount of space such as 7 days—or maybe even 10 days, just to be safe.

Within Syntax Statements

The following batch job would accomplish this:

```
:JOB WEEKLY,SCOPE.SYS,SCOPE
:RUN UTILITY
LOG LOGGLOB.SCOPE.SYS
DETAIL OFF
SCAN
RESIZE GLOBAL      EMPTY=10 MAYBE
RESIZE APPLICATION EMPTY=10 MAYBE
RESIZE PROCESS     EMPTY=10 MAYBE
RESIZE DISC        EMPTY=10 MAYBE
EXIT
:EOJ
```

Specifying MAYBE instead of YES avoids any resizing operations if 10 or more days of empty space currently exist in any log files.

Note that the maximum file size defaults to the current maximum file size for each file. This allows the files to be resized to new maximum sizes without affecting this weekly batch job.

RESIZE Command Reports

One standard report is produced when you resize a raw log file. It shows the three interrelated disc space categories of maximum file size, data records, and empty space, before and after resizing.

```
RESIZE GLOBAL DAYS=120;EMPTY=10
EMPTY SPACE RAISED TO MATCH FILE SIZE AND DATA RECORDS

FINAL RESIZING PARAMETERS:
FILE: LOGGLOB.SCOPE.SYS              MEGABYTES / DAY: 0.101199
               -----CURRENTLY-----   --AFTER RESIZING---
MAXIMUM SIZE:   65 DAYS (  6.6 MB)   120 DAYS ( 12.1 MB)  83% INCREASE
DATA RECORDS:   61 DAYS (  6.2 MB)    61 DAYS (  6.2 MB) NO DATA REMOVED
EMPTY SPACE:     4 DAYS (  0.5 MB)    59 DAYS (  6.0 MB) 1225% INCREASE
```

The megabytes/day value is used to convert between days and megabytes. It is either the value obtained during the SCAN function or a default for the type of data being resized.

The far right-hand column is a summary of the net change in each category of log-file space. Maximum size and empty space can increase, decrease, or remain unchanged. Data records have either no data removed or a specified amount of data removed during resizing.

If the RESIZE is done interactively and one or more parameters are defaults, you can get a preliminary resizing report. This report summarizes the current log file contents and any parameters that were provided. The report is provided to aid in answering questions on the unspecified parameters.

```
RESIZE GLOBAL DAYS=20

FILE RESIZING PARAMETERS (BASED ON AVERAGE DAILY
SPACE ESTIMATES AND USER RESIZING PARAMETERS)
FILE: LOGGLOB.SCOPE.SYS              MEGABYTES / DAY: 0.101199
                 -----CURRENTLY-----   --AFTER RESIZING---
MAXIMUM SIZE:   65 DAYS (  6.6 MB)    20 DAYS (  2.0 MB)
DATA RECORDS:   61 DAYS (  6.2 MB)    ??
EMPTY SPACE:     4 DAYS (  0.5 MB)    ??
```

In this example you would be prompted to supply the amount of empty space for the file before the final resizing report is given. If no action parameter is given for interactive resizing, you are prompted for whether or not to resize the log file immediately following the final resizing report.

Sample RESIZE Task

The following resizing task resizes the raw PROCESS log file:

`LOGFILE LOGGLOB.SCOPE.SYS`	(*The name of the log file to be resized.*) The default log file, LOGGLOB.SCOPE.SYS, is specified in this case.
`DETAIL=OFF`	(*The scan report will not give full detail.*)
`SCAN`	(*LOGGLOB.SCOPE.SYS will be read and a summary report on its contents will be produced.*) The scan is performed before the resize to increase the accuracy of the number-of-days calculations.

```
RESIZE PROCESS DAYS=60 EMPTY=30 YES
```

`DAYS=60`	(*Specifies holding a maximum of 60 days of data.*)
`EMPTY=30`	(*Specifies that 30 days of this file are currently empty.*) That is, the file is resized with no more than 30 days of data in the file to leave room for 30 more days out of a total of 60 days of space.

SCAN

`SCAN` [*logfile*]

The following related commands are discussed briefly in this section and in more detail elsewhere in this chapter under the command name: LIST, START, STOP, DETAIL, NOTES, and ALARMS.

The SCAN command reads a log file and writes a summary report on its contents. The format of this report depends on the commands issued prior to SCAN. For example,

- The LOGFILE and SCAN commands can create a report on the log file's disc space.

  The DETAIL, ALARMS, and NOTES commands can control the amount of information reported during this process.
- Default details can be disabled by turning off DETAIL and NOTES before performing a SCAN to produce a summary report on the contents of a log file.
- By enabling DETAIL before scanning a log file, you can produce a report on the changes in collection parameters over time.

  This report would show summaries of application and process collection whenever appropriate changes in collection occurred. The report also would tell you each time the SCOPE(XL) collector was restarted and the length of time the collector was not running.
- Process performance alarms against a log file and report when alarms would be satisfied. Alarm definitions must have been processed by the most current PARMFILE command in order to be processed during the SCAN. See the chapter on Performance Alarms for more details.

Within Syntax Statements

The SCAN command requires that a log file is opened. The log file to be scanned is the first of one of the following:

1. The log file named in the SCAN command itself.
2. The last log file opened by any previous command.
3. The default log file (LOGGLOB.SCOPE.SYS).

   In this case, interactive users are prompted to override the default log file name if so desired.

The following commands affect the operation of the SCAN function:

`LIST`	(*Redirects the report listing to another device.*) The default is to list to the standard list device.
`START`	(*Specifies the starting date and time of the scan.*) The default is the beginning of the log file.
`STOP`	(*Specifies the ending date and time of the scan.*) The default is the end of the log file.
`DETAIL`	(*Specifies the amount of detail in the report.*) The default (DETAIL=ON) specifies full detail.
`NOTES`	(*Specifies whether you want to see user and system notes found in the log file.*) The default is NOTES=ON.
`ALARMS`	(*Specifies whether or not you want to see individual Performance Alarms in the detailed report.*) The default is ALARMS=ON, if alarms are defined.

The report from the SCAN command consists of several parts. You can control whether some parts are present in the report by using the UTILITY commands listed above.

Sample SCAN Tasks

Scan the current Performance Collection Software raw log files, and display a report on the logging details.

This report includes details on the initial PARM file settings plus any changes to these settings. An overall summary of disc space consumed by each type of logging is also displayed. In this sample, the listing is defaulted to the job or session $STDLIST device.

`LOGFILE LOGGLOB.SCOPE.SYS`	(*Opens the LOGGLOB.SCOPE.SYS log file.*)
`DETAIL=ON`	(*Specifies full detail for the SCAN report.*)
`SCAN`	(*Reads LOGGLOB and produces a summary report on its contents.*)

SCOPE

```
        ┌ STATUS            ┐
        │ KILL              │
SCOPE   │ NEWPARM           │
        └ NOTE [message]    ┘
```

The SCOPE command allows you to communicate with the SCOPE(XL) data collector as it runs. If the Performance Collection Software data collector is not running *and* logged on to the SCOPE.SYS group, all SCOPE commands indicate that SCOPE(XL) is not running and the commands fail.

Note To issue a command to the SCOPE(XL) data collector, you must have SM or OP capability and execute access to files in the SCOPE.SYS group.

If the SCOPE(XL) collector is running, the SCOPE command looks for commands once a minute. You may have to wait as long as 60 seconds for SCOPE(XL) to recognize and act on any command you issue.

Commands that expect responses back from SCOPE(XL) wait until the response is received (or no more than 90 seconds), during which time you cannot enter additional UTILITY commands.

If the user running UTILITY does not have write and save access to files in their logon group, it may be impossible to retrieve information. A local response file called SCOPEOUT is created to receive data from SCOPE(XL). If two users in the same group attempt to receive data from SCOPE(XL) at the same time, the first user to create the SCOPEOUT file succeeds. The second user fails and will get a file-create error on this file.

SCOPE

The SCOPE command without a parameter quickly indicates whether SCOPE(XL) is running. (This command may be executed by users without SM or OP capability).

SCOPE STATUS

SCOPE with the STATUS parameter asks the currently executing SCOPE(XL) to return its version number and data collection information.

Most of this report is similar to part of the SCAN report (initial PARM file global information). Also, a report on the amount of disc space used in the four main log files—LOGGLOB, LOGAPPL, LOGPROC, and LOGDISC—is returned.

SCOPE KILL

SCOPE with the KILL parameter asks the currently executing SCOPE(XL) to terminate in an orderly fashion. UTILITY will wait until SCOPE(XL) terminates before continuing.

If you issue the SCOPE KILL command interactively, you are asked to verify that you actually want to kill SCOPE(XL)—you are given a chance to change your mind.

Batch execution of the SCOPE KILL command requires no further input.

Sample SCOPE KILL Task

You can use the `SCOPE KILL` command to stop SCOPE(XL):

```
SCOPE KILL
```

You are asked to verify this command:

```
KILL THE SCOPE COLLECTOR, ARE YOU SURE?
```

Respond with `YES` to kill SCOPE(XL).

You can also purge the RUN.SCOPE.SYS file to stop SCOPE(XL):

```
PURGE RUN.SCOPE.SYS
```

SCOPE NEWPARM

SCOPE with the NEWPARM parameter instructs SCOPE(XL) to reprocess its PARM file. Any changes found in the PARM file are acted upon without stopping and restarting the data collection job. (You cannot change the system ID at this time.)

SCOPE NOTE

SCOPE with the NOTE parameter places a user note into the Performance Collection Software global log file. Such notes may be viewed by performing a SCAN with NOTES=ON or NOTES=100. If no message is entered, you are prompted to enter a message (in interactive mode), or the job terminates (in batch mode).

Sample SCOPE NOTE Task

You can add a message to the Performance Collection Software log file. SCOPE(XL) must be running in order to execute this command.

```
SCOPE NOTE This will be a user-defined note.
```

SHOW

```
SHOW [ALL]
```

The SHOW command lists the names of the files that are open and the status of the UTILITY parameters that can be set, for example,

```
SHOW
  LOGFILE:  LOGGLOB.SCOPE.SYS
  PARMFILE: NOT OPENED
  DETAIL=ON for SCAN and PARMFILE functions
  NOTES=ON for the SCAN function
  LIST OUTPUT GOES TO "$STDLIST"
  DEFAULT STARTING DATE & TIME=08/11/89  6:16 PM
  DEFAULT STOPPING DATE & TIME=12/31/99 12:00 AM
```

Adding the optional parameter `ALL` prints more information about the log file if one is open, for example,

```
SHOW ALL
  GLOBAL      FILE: LOGGLOB.SCOPE.SYS        VERSION B
  APPLICATION FILE: LOGAPPL.SCOPE.SYS
  PROCESS     FILE: LOGPROC.SCOPE.SYS
  DISC SPACE  FILE: LOGDISC.SCOPE.SYS
  INDEX       FILE: LOGINDX.SCOPE.SYS
  SYSTEM ID: COOKIE

  DATA COLLECTOR: XL B.02.00
  DATA COVERS:   132 DAYS FROM 01/03/90 TO 05/14/90
  GLOBAL APPLICATION PROCESS DISCSPACE DATA RECORDS ARE AVAILABLE

  PARMFILE: NOT OPENED
  DETAIL = ON  for SCAN and PARMFILE functions
  NOTES  = ON  for the SCAN function
  LIST OUTPUT GOES TO "$STDLIST"
  THE DEFAULT STARTING DATE & TIME = 08/11/89  6:16 PM
  THE DEFAULT STOPPING DATE & TIME = 12/31/99 12:00 AM
```

START

$$\texttt{START} \left[\begin{array}{l} \textit{date} \ [\textit{time}] \\ \texttt{TODAY} \ [\textit{-days}]\,[\textit{time}] \end{array}\right]$$

Use the START command to set a starting date and time for the SCAN function. The default starting date is the date of the earliest record of any type in the log file.

The date format depends on the native language configured on the HP 3000 system being used. If you do not use Native Languages or have set the default language (NATIVE-3000), the date format is *mm/dd/yy* (month/day/year) such as 02/28/88 for February 28, 1988.

The time format also depends on the native language being used. For NATIVE-3000 the format is *hh:mm* AM or *hh:mm* PM (hour:minute in 12-hour format with the AM/PM suffix) such as 07:00 AM as 7 o'clock in the morning.

Note If you are not sure whether NLS is installed on your system, you can force UTILITY to use the NATIVE-3000 date and time formats by issuing the following statement before running UTILITY:

```
:SETJCW NLUSERLANG=0
```

If the date or time is entered in an unacceptable format, you are prompted with an example in the correct format.

If no start time is given, midnight (12:00 AM) is assumed. A starting time of midnight for a given day starts at the *beginning* of that day (00:00 on a 24-hour clock).

The keyword TODAY may be used to represent the current date. TODAY-*days* specifies the number of days prior to today's date. For example, TODAY-1 indicates yesterday's date.

STOP

$$\texttt{STOP}\left[\begin{array}{l} \textit{date}\ [\,\textit{time}\,] \\ \texttt{TODAY}\ [\,\text{-}\textit{days}\,]\,[\,\textit{time}\,] \end{array}\right]$$

The STOP command can be used to set the date and time for terminating the SCAN function. The default stopping date and time is the last date and time in the log files.

The formats for the STOP command are the same as for the START command and depend on the native language being used.

If no stop time is given, midnight (12:00 AM) is assumed. A stopping time of midnight for a given day stops at the *end* of that day (24:00 on a 24-hour clock).

The keyword TODAY may be used to represent the current date. TODAY-*days* specifies the number of days prior to today's date. For example, TODAY-1 indicates yesterday's date.)

VERSIONS

VERSIONS [*group*]

The VERSIONS command attempts to list the version numbers for all HP LaserRX/MPE and Performance Collection Software host programs and libraries. It also checks for the HP Cooperative Services programs used by Performance Collection Software to communicate with the PC software.

If no group name is supplied, the Performance Collection Software host programs are expected to be in the same group as UTILITY. The HP Cooperative Services programs are always expected to be in the PPC.SYS group.

```
Versions
LaserRX host programs and library versions in group SCOPE.SYS:

SCOPEXL   B.03.00.61
SCOPEXL2  B.00.00.06
EXTRACT   B.00.00.10
UTILITY   B.00.00.08
XL        HCS_REQUESTOR_LIBRARY B.00.00

HP Cooperative Services data communications programs
in group PPC.SYS:

HCS
HCSERVER
HP        HP32571A rev.(A.01.12N) Cooperative Services
```

5

Data Management: EXTRACT

This chapter details the EXTRACT commands. It includes a syntax summary and a command reference section that lists the commands in alphabetical order.)

Note Commands and parameters for EXTRACT can be entered with any combination of uppercase and lowercase letters. Only the first three letters of the command's name are required. For example, the command `APPLICATION DETAIL` can be abbreviated as `APP DET`.

Examples of various tasks using the EXTRACT program can be found in the programs's online HELP facility.

EXTRACT Command Syntax Summary

This following table contains a summary of EXTRACT commands syntax and their parameters.

Table 5-1. EXTRACT Commands: Syntax and Parameters

Command	Parameter
APPLICATION	ON DETAIL SUMMARY BOTH OFF
AUTO	
CONFIGURATION	ON OFF
DISC	ON OFF
EXIT E	
EXPORT	
EXTRACT	
GLOBAL	ON DETAIL SUMMARY BOTH OFF
HELP	*topic*
LIST	*device*
LOGFILE	*logfile*

Table 5-1.
EXTRACT Commands: Syntax and Parameters (continued)

Command	Parameter
MENU	
MONTHLY	*yymm* *mm*
MPE :	*command*
OUTPUT	*output filename* ,NEW ,PURGE ,APPEND
PROCESS	ON OFF
REPORT	[*reportfile*] [*,SHOW*]
SHIFT	*starttime - stoptime* ALL DAY NOWEEKENDS
SHOW	ALL
START	*date* [*time*] TODAY [-*days*] [*time*]
STOP	*date* [*time*] TODAY [-*days*] [*time*]
WEEKLY	*yyww* *ww*
YEARLY	*yyyy* *yy*

APPLICATION

```
             ┌ ON      ┐
             │ DETAIL  │
APPLICATION  │ SUMMARY │
             │ BOTH    │
             └ OFF     ┘
```

The APPLICATION command selects the type of application data that will be extracted or exported.

The default is `APPLICATION OFF`.

DETAIL or ON

The DETAIL or ON parameter specifies that raw, 5-minute detail data should be extracted.

When using the HP LaserRX/MPE Analysis software, detail data must be included in an extracted file in order to zoom on application details or use the file to draw application graphs with points every 5 minutes.

SUMMARY

The SUMMARY parameter specifies that raw data should be summarized into one data point per hour before being extracted. Summarization reduces the size of the application data to about one-tenth of the size of the detail data. Summarized data will be graphed more quickly since it takes fewer data records to produce a graph.

If summary data is not placed into the extracted file, hourly summaries can still be graphed, but the summaries will be generated from the detail data each time a graph is drawn.

If only application summaries are extracted, HP LaserRX/MPE application graphs with data points every 5 minutes cannot be drawn, and application zoom will return no data.

BOTH

The BOTH parameter specifies that 5-minute detail data and hourly summary data are to be extracted.

This option maintains the speed of access gained with hourly summary records, while allowing HP LaserRX/MPE application graphs with points every 5 minutes.

The disc space required to extract both detail and summary data is about 10 percent more than the space needed for extracted detail data alone.

OFF

The OFF parameter specifies that no application data is to be extracted.

AUTO

```
AUTO
```

The AUTO command starts automatic data extraction. No further interaction with EXTRACT is allowed. Commands entered before the AUTO command are honored, if possible. You can enter the default starting data for an extraction by using the run-time ;PARM= parameter. See the START command for more information.

If not previously specified, the LOGFILE and OUTPUT commands will assume the following defaults when the AUTO command is executed:

LOGFILE:	`LOGGLOB.SCOPE.SYS`
OUTPUT:	`RXLOG,APPEND`

The settings or defaults for all other parameters are used. For details on their actions, see APPLICATION, DISC, GLOBAL, PROCESS, SHIFT, START, and STOP.

The EXTRACT program terminates when the AUTO command is completed.

Sample AUTO Task

In this task, you perform an automatic data extraction.

Extract the last 7 days of global detail data from the raw log files into the RXLOG file. If RXLOG already exists, append the new data to it. When the extraction is finished, end EXTRACT.

```
:RUN EXTRACT;PARM=7
AUTO
```

Note The AUTO command is included to ensure compatibility with earlier versions of the EXTRACT program. If possible, use the more powerful commands: WEEKLY, MONTHLY, and YEARLY.

CONFIGURATION

```
              ┌     ┐
              │ ON  │
CONFIGURATION │     │
              │ OFF │
              └     ┘
```

The CONFIGURATION command specifies whether or not to export system configuration information.

The default is `CONFIGURATION OFF`.

All configuration information collected between the start and stop dates is exported. Shift times and NOWEEKENDS parameters are ignored.

Note The CONFIGURATION command affects only the EXPORT function. It does not affect the EXTRACT function since the EXTRACT function *always* extracts system configuration information.

DISC

$$\texttt{DISC} \begin{bmatrix} \texttt{ON} \\ \texttt{OFF} \end{bmatrix}$$

The DISC command specifies whether or not to extract or export disc space information.

The default is `DISC OFF`.

Specifying DETAIL, SUMMARY, or BOTH is the same as specifying ON, since disc space data is only collected once a day.

All disc space data collected between the start and stop dates is extracted. Shift times are ignored, but the NOWEEKENDS parameter *is* honored.

EXIT

```
EXIT
E
```

The EXIT command ends the EXTRACT program. (If you are using terminal softkeys, you can press (F8)).

EXPORT

```
EXPORT
```

The EXPORT command starts the process of exporting data. If not previously specified, the LOGFILE and REPORT commands will assume the following defaults when the EXPORT command is executed:

LOGFILE	`LOGGLOB.SCOPE.SYS`
REPORT	`REPTFILE.SCOPE.SYS`

See the discussion of the EXPORT function later in this chapter for more information on how to export files.

The settings or defaults for all other parameters are used. For details on their actions, see APPLICATION, CONFIGURATION, DISC, GLOBAL, PROCESS, SHIFT, START, and STOP.

Note The OUTPUT command is not used by the EXPORT command. EXPORT can create up to seven different files based on the types of data and level of summarization selected.

Logon group files having the following names will be purged when the EXPORT function is executed. These files will be recreated, if needed.

`XFERGLOB`	Global Detail Data file.
`XFERGSUM`	Global Summary Data file.
`XFERAPPL`	Application Detail Data file.
`XFERASUM`	Application Summary Data file.
`XFERPROC`	Process Detail Data file.
`XFERDISC`	Disc Space Data file.
`XFERCONF`	Configuration Data file.

Sample EXPORT Task

In this task, you export raw log file data collected yesterday (8:00 AM to 5:00 PM) using the default report file.

```
:RUN EXTRACT
LOGFILE LOGGLOB.SCOPE.SYS
START TODAY-1 8:00 AM
STOP  TODAY-1 5:00 PM
GLOBAL SUMMARY
REPORT REPTFILE.SCOPE.SYS
EXPORT
```

EXTRACT

```
EXTRACT
```

The EXTRACT command starts data extraction. If not previously specified, the LOGFILE and OUTPUT commands will assume the following defaults when the EXTRACT command is executed:

LOGFILE	`LOGGLOB.SCOPE.SYS`
OUTPUT	`RXLOG,NEW`

The settings or defaults for all other parameters are used. For details on their actions, see APPLICATION, DISC, GLOBAL, PROCESS, SHIFT, START, and STOP.

Sample EXTRACT Task 1

In this task, you extract data from raw log files using EXTRACT's default settings:

1. Extract data from the raw log files (LOGGLOB.SCOPE.SYS).
2. Extract the last 30 full days of global detail data from the log file.

```
:PURGE RXLOG
:RUN EXTRACT
EXTRACT
EXIT
```

Sample EXTRACT Task 2

In this task, you copy data from one extracted log file (RXJAN) to another (RXSUM), and summarize global detail data into hourly summary data. Assume that `:RUN EXTRACT` has been completed.

```
LOGFILE RXJAN
OUTPUT  RXSUM
GLOBAL  SUMMARY
EXTRACT
```

Sample EXTRACT Task 3

In this task, you append data from raw log files to the existing RXSUM extracted log file. Original extraction specifications from the RXSUM file are maintained. Extracting data is stopped on February 28, 1990.

```
LOGFILE LOGGLOB.SCOPE.SYS
OUTPUT RXSUM,APPEND
STOP 2/28/90
EXTRACT
```

Sample EXTRACT Task 4

In this task, you create a new extracted log file called RXJAN90.

Purge any existing file with this name. Extract all data, detail, and summaries from the raw log files from January 1, 1990 to January 31, 1990.

```
LOGFILE LOGGLOB.SCOPE.SYS
OUTPUT RXJAN90,PURGE
START 1/1/90
STOP  1/31/90
GLOBAL BOTH
APPLICATION BOTH
PROCESS BOTH
DISC DETAIL
EXTRACT
```

Sample EXTRACT Task 5

In this task, you extract data from the raw log files for January 1, 1901 to December 31, 1999.

Extract only global and application summary data from 8:00 AM to 5:00 PM, excluding Saturday and Sunday. The LOGFILE command defaults to LOGGLOB.SCOPE.SYS, so it is not provided.

Within Syntax Statements

```
OUTPUT ALLDATA,PURGE
START 1/1/1
STOP 12/31/99
SHIFT 8:00 AM - 5:00 PM NOWEEKENDS
GLOBAL SUMMARY
APPLICATION SUMMARY
EXTRACT
```

GLOBAL

```
        ┌ ON      ┐
        │ DETAIL  │
GLOBAL  │ SUMMARY │
        │ BOTH    │
        └ OFF     ┘
```

The GLOBAL command selects how much global data is extracted or exported.

The default is `GLOBAL DETAIL`.

DETAIL

The DETAIL parameter specifies that raw detail collected at 5-minute intervals is to be extracted.

Detail data must be placed into an extracted file before you can draw any HP LaserRX/MPE global graphs with points every 5 minutes.

SUMMARY

The SUMMARY parameter specifies that raw data should be summarized hourly before being extracted.

Summarization reduces the size of the global data to about one-twelfth that of the detail data.

Summarized data will be graphed on HP LaserRX/MPE Analysis Software more quickly since it takes fewer data records to produce a graph.

If summary data is not placed into the extracted file, then hourly summaries can be graphed, but the summaries are generated from the detail data each time a graph is drawn.

HP LaserRX/MPE Global graphs cannot be drawn with data points every 5 minutes if only global summaries are extracted.

BOTH

The BOTH parameter specifies that 5-minute detail data *and* hourly summary data are to be extracted.

This option maintains the speed of access gained with the hourly summary records while permitting you to draw HP LaserRX/MPE Global graphs with points every 5 minutes.

The disc space required to extract both detail and summary data is about 8 percent more than the space needed for extracted detail data alone.

OFF

The OFF parameter specifies that no global data is to be extracted.

Note This option is not recommended with the current Performance Collection Software product since you must have global data to properly understand overall system behavior. HP LaserRX/MPE Global graphs cannot be drawn unless the extracted file contains at least one type of global data.

HELP

```
HELP [topic]
```

Enter the HELP command to access the online HELP facility interactively. This facility and the MPE HELP facility work in the same way. You can enter parameters to obtain information on EXTRACT commands and tasks, and on the HELP command.

If you are using a terminal with HP softkey support, you can get help on the functions of a softkey by pressing the softkey while you are in the HELP subsystem. Pressing F8 (`Exit HELP`) exits the HELP subsystem and returns you to the EXTRACT program, or you can exit the HELP system by typing `EXIT` or `STOP`.

You can also request help on a specific topic. For example, type the following:

```
HELP TASKS
```

or

```
HELP SHIFT
```

When you use this form of the HELP command, you will receive the help text for the specified topic and remain in the EXTRACT command entry context. Since you do not enter the HELP subsystem interactively, you will not have to exit or stop before entering the next EXTRACT command.

LIST

```
LIST [device]
```

You can change the list device for all EXTRACT reports in two ways:

- To issue a file equation before running the EXTRACT program, enter

  ```
  :FILE RXLIST;DEV=LP
  ```

  or

- Once EXTRACT is running, enter

  ```
  LIST LP
  ```

The device parameter on the LIST command must represent a valid configured device on the current HP 3000 system. It should be a device that can be written to (normally, this would be a printer). If you need to specify more than just the device name, you can issue a file equation for file RXLIST, then enter the device by using the LIST command.

```
MPE FILE RXLIST;ENV=LP2.HPENV.SYS
LIST HP2680
```

To return the listing device back to the standard list device, enter

```
LIST TERM
```
or
```
LIST $STDLIST
```
or
```
LIST *
```

To determine the current list device, enter the LIST command without parameters as follows:

```
LIST
```

If the LIST device is not $STDLIST, most commands are echoed as they are entered.

LOGFILE

`LOGFILE` [*logfile*]

A log file must be opened for all EXTRACT program functions. You can do this explicitly, by issuing the LOGFILE command, or implicitly, by issuing the EXTRACT or AUTO command. If a log file name is not provided, the default LOGGLOB.SCOPE.SYS is used.

You can specify the name of either a raw or extracted log file. You cannot specify the name of a file created by the EXPORT command. If you specify an *extracted* log file name, all information is obtained from this single file. If you specify a *raw* log file name, you must specify the name of the global log file before you can access the raw log file. This is the *only* raw log file name you should specify.

It is assumed that the other raw log files are in the same group as the global log file. They have the following names:

`LOGGLOB`	Global log file.
`LOGAPPL`	Application log file.
`LOGPROC`	Process log file.
`LOGDISC`	Disc Space log file.
`LOGINDX`	Index file (MPE/iX only).

The general contents of the log file are displayed when the log file is opened.

You can use a file equation to open an *extracted* file, but you should not use file equations to access *raw* log files.

You must create a separate group for each set of raw log files that you keep on any one HP 3000. (The group and account do not have to be SCOPE.SYS.)

Caution Do not *rename* raw log files! When accessing these files, it is assumed that the standard log file names are in effect. For example, if you must rename log files to place log files from multiple systems on the same HP 3000 for analysis, you should first *extract* the data, then rename the extracted log files.

The following is an example of raw log files that have been opened in SCOPE.SYS:

```
GLOBAL FILE:      LOGGLOB.SCOPE.SYS VERSION B
APPLICATION FILE: LOGAPPL.SCOPE.SYS
PROCESS FILE:     LOGPROC.SCOPE.SYS
DISC SPACE FILE:  LOGDISC.SCOPE.SYS
INDEX FILE:       LOGINDX.SCOPE.SYS
SYSTEM ID: COOKIE PTC Hewlett-Packard SERIES 967

DATA COLLECTOR: XL B.02.00
DATA COVERS: 129 DAYS FROM 01/03/90 TO 05/11/90
GLOBAL APPLICATION PROCESS DISCSPACE DATA RECORDS ARE AVAILABLE

THE FIRST GLOBAL      RECORD IS ON 01/03/90 AT 12:40 PM
THE FIRST APPLICATION RECORD IS ON 01/03/90 AT 12:40 PM
THE FIRST PROCESS     RECORD IS ON 01/22/90 AT 10:42 AM
THE FIRST DISC SPACE  RECORD IS ON 08/11/89 AT 6:16 PM
```

You can verify the log file that you have opened with the SHOW command described later.

You can open another log file at any time by entering another LOGFILE command. Any currently open log file is closed before the new log file is opened.

MENU

```
MENU
```

The MENU command prints a short list of the available EXTRACT commands.

`HELP`	Get information on commands and options.
`LOGFILE`	Specify a log file to be analyzed.
`OUTPUT`	Specify a destination file for extraction.
`START`	Specify a starting date and time for EXTRACT.
`STOP`	Specify an ending date and time for EXTRACT.
`SHIFT`	Specify starting and stopping times for each day.
`GLOBAL`	Extract global records (DETAIL/SUMMARY/BOTH/OFF).
`APPLICATION`	Extract application records (DETAIL/SUMMARY/BOTH/OFF).
`PROCESS`	Extract process records (DETAIL/OFF).
`DISC`	Extract disc space records (DETAIL/OFF).
`LIST`	Specify the listing device for EXTRACT.
`SHOW`	Show the current program settings.
`EXTRACT`	Copy selected records to output (or append) file.
`WEEKLY`	Extract the current week's data into an automatically named file.
`MONTHLY`	Extract the current month's data into an automatically named file.
`YEARLY`	Extract the current year's data into an automatically named file.
`MPE` *or* `:`	Execute an MPE command.
`MENU`	List the command menu.
`EXIT` *or* `E`	Terminate this program.
`CONFIG`	Export CONFIGURATION records (ON/OFF).
`REPORT`	Specify a REPORT file format for EXPORT.
`EXPORT`	Copy log file records to HOST format files.

MONTHLY

$$\texttt{MONTHLY} \begin{bmatrix} yymm \\ mm \end{bmatrix}$$

The MONTHLY command performs data extraction based on a calendar month.

When you execute this command, the start and stop dates are set to the proper dates based on the month and year of the data extracted.

The name of the output file consists of the letters RXMO followed by the last two digits of the year and the two-digit number of the month being extracted. For example, March 1990 would be output to a file named RXMO9003.

Enter one of the following:

`MONTHLY`	To extract data from the current (default) month.
`MONTHLY`*mm*	To extract data for a specific month from this year's data (where *mm* is a number from 01 to 12).
`MONTHLY`*yymm*	To extract data for a specific month *and* year (where *yymm* is a single number composed of the last two digits of the year and the two-digit month number). For example, February 1991 would be `MONTHLY 9102`.

If you do not specify the log file before executing the MONTHLY command, it defaults to the file LOGGLOB.SCOPE.SYS.

The type of data extracted and the level of summarization of that data follow the normal rules for EXTRACT and can be set before executing the MONTHLY command. These settings are honored unless the monthly output file already exists. If it does, data is appended to it based on the original type of data selected.

The MONTHLY command has a special feature. It opens the *previous* month's extracted file and checks to see if it is filled—whether it contains data extracted up to the last day of the month. If not, the MONTHLY command appends data to this file to complete the previous month's extraction.

For example, a MONTHLY command is executed on May 7, 1991. This creates a log file named RXMO9105 containing data from May 1 through the current date (May 7).

On June 4, 1991, another MONTHLY command is executed. Before the RXMO9106 file is created for the current month, the RXMO9105 file from the previous month is opened. When it is found to be incomplete, data is appended to it to complete the extraction through May 31, 1991. Then, the RXMO9106 file is created to hold data from June 1, 1991 to the current date (June 4).

As long as you execute the MONTHLY command at least once a month, this feature completes each month's files before the next month's file is created. Whenever you see two monthly files adjacent to each other—for example, RXMO9005 and RXMO9006—you can assume safely that the first file is complete for that month, and it can be archived and purged.

MPE

`MPE` *or* : [*command*]

You can enter an MPE command without exiting EXTRACT by entering `MPE` or a colon (:) followed by a valid MPE V or MPE/iX command. For example,

```
MPE SHOWJOB
:TELL MANAGER.SYS; HI
:EDITOR
MPE RUN FCOPY.PUB.SYS
```

The following commands *cannot* be executed in this way:

```
ABORT      DO      HELLO      SETCATLOG
BYE        EOD     JOB        SHOWCATALOG
CHGROUP    EOJ     LISTREDO   REDO
DATA       EXIT    OPTION     RESUME
```

UDC, COMMAND FILE, and implicit RUN commands can be executed on MPE/iX but not on MPE V. The explicit RUN command is allowed on both MPE V and MPE/Ix.

If you omit the command, you will be prompted to supply it.

```
MPE
Enter MPE Command: SHOWJOB
```

OUTPUT

```
                              ┌          ┐
                              │ ,NEW     │
OUTPUT [ output filename ]    │ ,PURGE   │
                              │ ,APPEND  │
                              └          ┘
```

Use the OUTPUT command to open and, if necessary, to create an extracted log file in which to write extracted SCOPE(XL) records. If you do not provide an output file name, EXTRACT assumes it is RXLOG.

The optional second parameter specifies the action to be taken if an output file with the same name exists.

`NEW`	(*Specifies that the output file must be a new file.*) This is the default action in batch mode. If a file with the same name exists, the batch job terminates.
`PURGE`	(*Specifies that any existing file should be purged to make room for the new output file.*)
`APPEND`	(*Specifies that an existing Performance Collection Software extracted file has data appended to it.*) If no file exists with the output file name specified, a new file is created.

If you do not specify an action, the default action NEW will be used (in batch mode), or you will be prompted to enter an action if a duplicate file is found (in interactive mode).

Note This command *is not* used with the EXPORT command.

PROCESS

$$\texttt{PROCESS} \begin{bmatrix} \texttt{ON} \\ \texttt{OFF} \end{bmatrix}$$

The PROCESS command selects whether or not process data will be extracted or exported.

The default is PROCESS OFF.

`ON`	(*Specifies that process detail data should be extracted.*) Process data must be placed into an extracted file in order to display tabular process data by using the *zoom process* function in the HP LaserRX/MPE Analysis Software.
`OFF`	(*The OFF parameter specifies that process data should not be extracted.*)

Note Process data can increase the size of an extracted log file significantly. If you plan to copy the log file to a PC for analysis, you might want to limit the amount of process data extracted.

REPORT

`REPORT` [*report file*] [*,SHOW*]

Use the REPORT command to select the definition file that will be used by the EXPORT function. The default report file is REPTFILE in the same group and account as the EXTRACT program (usually REPTFILE.SCOPE.SYS).

`,SHOW`	(*Specifies that the field positions and starting columns should be listed for all reports contained in the report file.*) This report may be used when export files are processed by other programs.

SHIFT

$$\texttt{SHIFT} \begin{bmatrix} \textit{starttime-stoptime} \\ \texttt{ALL DAY} \end{bmatrix} \begin{bmatrix}\texttt{NOWEEKENDS}\end{bmatrix}$$

The SHIFT command can be used to limit data extraction to certain hours of the day corresponding to work shifts and to exclude weekends (Saturday and Sunday). The default is ALL DAY (everyday including weekends).

The *starttime* and *stoptime* parameters are entered in the same format as the time in the START command. Shifts that span midnight are permitted. If *starttime* is scheduled *after* the *stoptime*, the shift will start at the start time and proceed past midnight, ending at the *stoptime* of the next day.

Specifying SHIFT ALL DAY selects the default shift of 12:00 AM–12:00 AM (or 00:00–24:00 on a 24-hour clock).

Specifying the NOWEEKENDS parameter discontinues data extraction on Saturdays and Sundays. If NOWEEKENDS is entered in conjunction with a shift that spans midnight, the weekend will be considered to consist of those shifts that *start* on Saturday or Sunday.

SHOW

```
SHOW [ALL]
```

The SHOW command lists the names of the opened files and the status of the EXTRACT parameters that can be set. For example:

```
SHOW
  LOGFILE:  LOGGLOB.SCOPE.SYS
  OUTPUT:   RXLOG.SCOPE.SYS           **NEW OUTPUT FILE**

  THE  DEFAULT  STARTING DATE & TIME = 04/16/90 12:00 AM
  THE  DEFAULT  STOPPING DATE & TIME = 05/16/90 12:00 AM
  THE  DEFAULT  SHIFT = 12:00 AM - 12:00 AM

  GLOBAL..... BOTH DETAIL & SUMMARY RECORDS WILL BE EXTRACTED
  APPLICATION...............SUMMARY RECORDS WILL BE EXTRACTED
  PROCESS..........DETAIL.......... RECORDS WILL BE EXTRACTED
  DISC SPACE.....................NO RECORDS WILL BE EXTRACTED
  CONFIGURATION....DETAIL...........RECORDS WILL BE EXPORTED
  LIST OUTPUT GOES TO "$STDLIST"
```

Within Syntax Statements

Adding the optional ALL parameter will cause more information about the log file to be printed. For example:

```
LOGFILE:  LOGGLOB.SCOPE.SYS
  GLOBAL      FILE: LOGGLOB.SCOPE.SYS
  APPLICATION FILE: LOGAPPL.SCOPE.SYS
  PROCESS     FILE: LOGPROC.SCOPE.SYS
  DISC SPACE  FILE: LOGDISC.SCOPE.SYS
  INDEX       FILE: LOGINDX.SCOPE.SYS
  SYSTEM ID: COOKIE PTC Hewlett-Packard SERIES 967

  DATA COLLECTOR: XL B.02.00
  DATA COVERS:   134 DAYS FROM 01/03/90 TO 05/16/90
  GLOBAL APPLICATION PROCESS DISCSPACE DATA RECORDS ARE AVAILABLE

OUTPUT:   RXLOG.SCOPE.SYS           **APPENDING**
  SYSTEM ID: COOKIE PTC Hewlett-Packard SERIES 967
  DATA COLLECTOR: XL B.02.00
  DATA COVERS:   28 DAYS FROM 02/01/90 TO 02/28/90
  SHIFT IS:       8:00 AM -  5:00 PM

START APPENDING AT END OF CURRENT DATA IN OUTPUT FILE
USER SELECTED STOPPING DATE & TIME = 03/31/90 12:00 AM
APPENDING TO  SHIFT =  8:00 AM -  5:00 PM

GLOBAL..... BOTH DETAIL & SUMMARY RECORDS WILL BE EXTRACTED
APPLICATION BOTH DETAIL & SUMMARY RECORDS WILL BE EXTRACTED
PROCESS..........DETAIL.......... RECORDS WILL BE EXTRACTED
DISC SPACE.......DETAIL.......... RECORDS WILL BE EXTRACTED
CONFIGURATION....DETAIL.......... RECORDS WILL BE EXPORTED

LIST OUTPUT GOES TO "$STDLIST"
```

START

$$\texttt{START}\left[\begin{array}{l} \textit{date}\ \left[\textit{time}\right] \\ \texttt{TODAY}\ \left[\textit{-day}\right]\left[\textit{time}\right] \end{array}\right]$$

Use the START command to set a starting date and time for the EXTRACT function. The default starting date is the date 30 full days before the last date in the log file *or* the date of the earliest record the log file, if less that 30 days are present.

You can control the default starting by using the ;PARM= parameter at run time. If this parameter is greater than zero, the default starting data is the specified number of days before the last date in the log file. For example, to extract the *last* 7 full days of data from the log file, type the following:

```
:RUN EXTRACT;PARM=7
```

The date format depends on the native language configured on the HP 3000 system being used. If you do not use Native Languages or you have set the default language (NATIVE-3000), the date format is mm/dd/yy (month/day/year) such as 02/28/91 for February 28, 1991.

The time format also depends on the native language being used. For NATIVE-3000, the format is *hh:mm* AM or *hh:mm* PM (hour:minute in a 12-hour format with the AM or PM suffix). For example, 07:00 AM is 7 o'clock in the morning.

Note If you are not sure whether NLS is installed on your system, you can force EXTRACT to use the NATIVE-3000 date and time formats by issuing the following statement before you run EXTRACT:

```
:SETJCW NLUSERLANG=0
```

If the date or time is entered in an unacceptable format, you are prompted with an example in the correct format.

Within Syntax Statements

If no start time is given, midnight (12:00 AM) is assumed. A starting time of midnight for a given day starts at the *beginning* of that day (00:00 on a 24-hour clock).

The keyword TODAY may be used to represent the current date. TODAY-*days* specifies the number of days prior to today's date. For example, TODAY-1 indicates yesterday's date.

STOP

$$\texttt{STOP}\left[\begin{array}{l}\textit{date}\ [\,\textit{time}\,] \\ \texttt{TODAY}\ [\,\textit{-day}\,]\,[\,\textit{time}\,]\end{array}\right]$$

The STOP command can be used to terminate the EXTRACT function on a specified date and time. The default stopping date and time is the last date and time recorded in the log files.

The formats for the STOP command are the same as those for the START command and depend on the language being used.

If no stop time is given, midnight (12:00 AM) is assumed. A stopping time of midnight for a given day stops at the *end* of that day (24:00 on a 24-hour clock).

The keyword TODAY may be used to represent the current date. TODAY-*days* specifies the number of days prior to today's date. For example, TODAY-1 indicates yesterday's date.

WEEKLY

$$\texttt{WEEKLY} \begin{bmatrix} yyww \\ ww \end{bmatrix}$$

The WEEKLY command specifies data extraction based on a calendar week. A week is defined as seven days starting on Monday and ending on Sunday.

When this command is executed, the start and stop dates are set to the proper dates based on the week and year of the extracted data.

The name of the output file consists of the letters RXWE followed by the last two digits of the year and the two-digit week number for the week being extracted. For example, the 20th week of 1990 (from Monday, May 14 to Sunday, May 20) would be output to a file named RXWE9020.

Enter one of the following:

`WEEKLY`	To extract the current week's data.
`WEEKLY` *ww*	To extract a specific week's data from this year's data (where *ww* is any number from 01 to 52).
`WEEKLY` *yyww*	To extract data for a specific week *and* year (where *yyww* is a single number composed of the last two digits of the year and the two-digit week-of-the-year number). For example, the 20th week of 1991 would be `WEEKLY 9120`.

If you do not specify the log file before executing the WEEKLY command, it defaults to the file LOGGLOB.SCOPE.SYS.

The type of data extracted and the level of summarization of that data follow the normal rules for EXTRACT and can be set before executing the WEEKLY command. These settings are honored unless the weekly output file already exists. If it does, data is appended to it based on the original type of data selected.

The WEEKLY command has a special feature. It opens the *previous* week's extracted file and checks to see if it is filled—whether it contains data extracted up to the last day of the week. If not, the WEEKLY command appends data to this file to complete the previous week's extraction.

For example, a WEEKLY command is executed on Thursday, May 17, 1991. This creates a log file named RXWE9120 containing data from Monday, May 14 through the current date (May 17).

On Wednesday, May 23, 1991 another WEEKLY command is executed. Before the RXWE9121 file is created for the current week, the RXWE9120 file from the previous week is opened. When it is found to be incomplete, data is appended to it to complete the extraction through on Sunday, May 20, 1991. Then, the RXWE9121 file is created to hold data from Monday, May 21, 1991 to the current date (May 23).

As long as you execute the WEEKLY command at least once a week, this feature completes each week's file before the next week's file is created. Whenever you see two weekly files adjacent to each other (for example, RXWE9120 and RXWE9121) you can assume safely that the first file is complete for that week, and it can be archived and purged.

Note The weeks are numbered based on their starting day. Thus week one of the year is the week starting on the first Monday of that year. Any days before that Monday belong to the last week of the previous year.

Sample WEEKLY Task

In this task, you extract the current week's data and complete last week's extracted file, if it is still present.

```
GLOBAL BOTH
APPLICATION BOTH
PROCESS DETAIL
DISC DETAIL
WEEKLY
```

A file named RXWE followed by the current year and week of the year is used as the output file.

YEARLY

$$\texttt{YEARLY} \begin{bmatrix} yyyy \\ yy \end{bmatrix}$$

The YEARLY command specifies a data extraction based on a calendar year. During execution, the command sets the start and stop dates to the proper dates, based on the year being extracted.

The name of the output file consists of the letters RXYR followed by the four digits of the year being extracted. Thus, data from 1991 would be output to a file named RXYR1991.

Enter one of the following:

`YEARLY`	To extract the current year's data.
`YEARLY` *yy*	To extract a specific year's data (where *yy* is a number from 00 to 99). The specifications 00 to 60 assume the years 2000 to 2060, whereas 61 to 99 assume the years 1961 to 1999.
`YEARLY` *yyyy*	To extract a specific year's data (where *yyyy* is the full-year numbered 1961 to 2060).

If you do not specify the log file before executing the YEARLY command, it defaults to the file LOGGLOB.SCOPE.SYS.

The type of data extracted and the level of summarization of that data follow the normal rules for EXTRACT and can be set before executing the YEARLY command. These settings are honored unless the yearly output file already exists. If it does, data is appended to it based upon the type of data selected originally.

The YEARLY command has a special feature. It opens the *previous* year's extracted file and checks to see if it is filled—whether it contains data extracted up to the last day of the year. If not, the YEARLY command appends data to this file to complete the previous year's extraction.

For example, a YEARLY command is executed on December 15, 1991. This creates a log file named RXYR1991 that contains data from January 1, 1991 to the current date (December 15).

On January 5, 1992, another YEARLY command is executed. Before the RXYR1992 file is created for the current year, the RXYR1991 file from the previous year is opened. When it is found to be incomplete, data is appended to it to complete its extraction until December 31, 1991. Then, the RXYR1992 file is created to hold data from January 1, 1992 to the current date (January 5).

As long as you execute the YEARLY command at least once a year, each year's file is completed before the next year's file is created. Whenever you see two yearly files adjacent to each other (for example, RXYR1990 and RXYR1991), you can assume safely that the first file is complete for that year, and it can be archived and purged.

Note The previous paragraph is true *only* if the raw log files are sized large enough to hold *one full year* of data. It would be more common to size the raw log files smaller and execute the YEARLY command more often (such as once a month).

Sample YEARLY Task

In this task, you append to the existing yearly summary file (or create it, if necessary).

Add application and global summary data plus disc detail only.

```
GLOBAL SUM
APPLICATION SUM
PROCESS OFF
DISC DETAIL
YEARLY
```

A file named RXYE followed by the current year is used as the output file.

Overview of the Export Action

EXPORT copies Performance Collection Software log file data into a form that can be easily accessed without passing through the HP LaserRX/MPE analysis programs. Any valid raw or extracted log file can be the source of this data.

The process is summarized in figure 5-1.

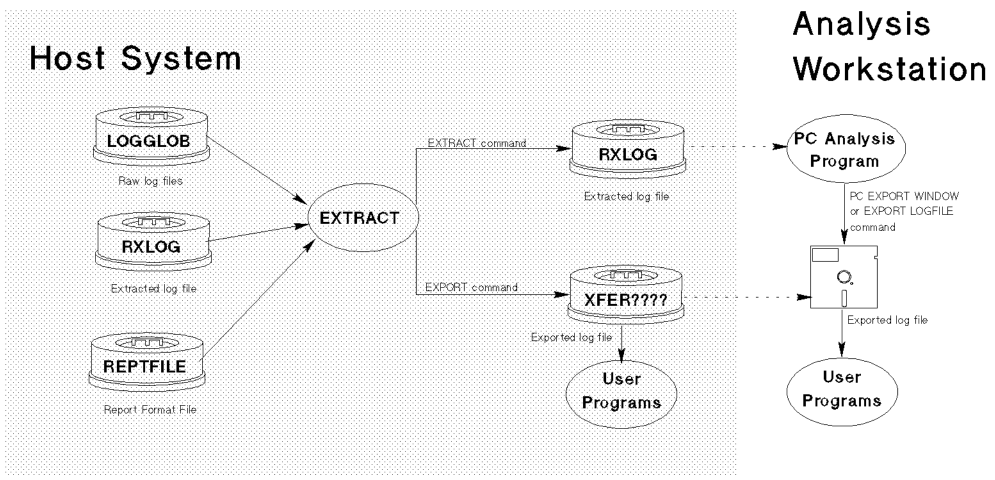

Figure 5-1. The Export Function

Exported files can be used in a variety of ways, such as custom graphics packages, databases, and user-written analysis programs. The main advantage of using exported files rather than the analysis programs to export the log file data lies in the significant time saved by not having to transport the data to the PC.

How to Export Data

In the simplest form, you can export data by specifying the default log file and the default report file, then starting the export. The default report file allows you to export files similar to the current HP LaserRX/MPE EXPORT LOGFILE function.

```
:RUN EXTRACT
LOGFILE LOGGLOB.SCOPE.SYS
REPORT REPTFILE.SCOPE.SYS
EXPORT
```

Exported data is in a file called XFERGLOB in a format suitable for loading into a spreadsheet.

If you want to export something other than this default set of data, you can use various commands and files in conjunction with the Export command.

- You can export the following types of data:

`Global`	*(Five-minute and hourly summaries.)*
`Application`	*(Five-minute and hourly summaries.)*
`Process`	*(One-minute details.)*
`Disc`	*(Daily data points.)*
`Configuration`	*(One record containing Performance Collection Software PARM file information, etc., for each time the collector started.)*

- You can specify what data items are needed for each type of data. Refer to the tables under "Data Items for Exporting Data" later in this section for a list of the data items for each type of data.
- You can choose to specify starting and ending dates for the data along with the SHIFT and WEEKEND EXCLUSION filters.
- You can specify the format of the exported data in an ASCII report file. This file can be created using any standard host editor program or you can use the default file, REPTFILE.

Sample Export Tasks

Several sample report files are furnished with the Performance Collection Software. They may be used initially to perform common reporting tasks or as a starting point for custom tasks.

- Generate a CPU and DISC report on a printer. The REPTHIST report file contains the specifications to generate a character graph of CPU and DISC usage for a system over time. This graph is composed of printable characters and may be printed on any device capable of 132 column printing. This example generates a graph of the last seven days on a system and should take approximately two pages (34 pages if five-minute detail is selected instead of hourly summaries).

```
:HELLO SCOPE.SYS,SCOPE
:RUN EXTRACT
REPORT REPTHIST
GLOBAL SUMMARY
START TODAY-7
EXPORT
EXIT
:COMMENT At this point the data is in a file XFERGSUM
:COMMENT Now copy it to the printer
:FILE PRINTER;DEV=LP
:FCOPY FROM=XFERGSUM;TO=*PRINTER
:PURGE XFERGSUM
```

- Summarize the process data and list the top CPU consumers for an interval. The PROCJOB job stream has been furnished to use the EXTRACT program and other standard MPE programs such as EDITOR, FCOPY and SORT to perform this task. It will print a report showing all processes which ran to completion yesterday, sorted by the amount of CPU each one used. The report file used by PROCJOB is called REPTPROC.

  To print the top processes on your system for yesterday:

```
:STREAM PROCJOB.SCOPE.SYS  (You may need to add passwords)
Examine the job's $STDLIST for the report
```

- Summarize the process data and list the top programs, as ranked by amount of CPU used. While similar to the previous task, this one adds the need to combine multiple executions of a program file, multiple processes, into a summary for the program file. A simple program has been written to combine multiple executions of a program into a single report line. The name of this program is TOPCPU and its source, in COBOL, is in the file TOPCPUS. The TOPJOB job stream will produce a report of the top programs which ran on your system yesterday. It uses the REPTTOP report file.

  To print the top programs on your system for yesterday:

  ```
  :STREAM TOPJOB.SCOPE.SYS  (you may need to add passwords)
  Examine the job's $STDLIST for the report
  ```

- Produce a customized export file. If one of the previous tasks is similar to what you desire, you can make a copy of the report file, then customize it. If you want to create a totally new export format then you can copy the report file REPTALL.SCOPE.SYS and modify it. The REPTALL file contains every possible item for each different data type so all you need to do is delete those items which are not of interest to you. This is easier than typing a report file from scratch.

Note Using the REPTALL report as furnished will probably cause warnings saying not all the data would fit into the maximum width data record. This is normal and simply indicates that you should select a subset of data to export.

Export Data Files

- Export can create up to seven exported files, depending on the types of data and the report file selected. A log file will not be created if items were not selected for the type of data in the report file or if the EXTRACT program did not receive the proper command prior to the EXPORT command.

  All seven data files will be purged at the beginning of an EXPORT function regardless of whether or not they will be recreated later. Be careful not to have other files with the following names in your logon group when performing the EXPORT function.

  The EXPORT log file names are:

`XFERGLOB`	Global Detail Data file.
`XFERGSUM`	Global Hourly Summary Data file.
`XFERAPPL`	Application Detail Data file.
`XFERASUM`	Application Hourly Summary Data file.
`XFERPROC`	Process Detail Data file.
`XFERDISC`	Disc Space Data file.
`XFERCONF`	Configuration Data file.

  You may use file equations to redirect any of these files to another disc file name. Redirection to nondisc files is not supported currently.

In summary, the following commands affect the actions of the Export function:

```
GLOBAL              START           LOGFILE
APPLICATION         STOP            REPORT
PROCESS             SHIFT
DISCSPACE
CONFIGURATION
```

For details on these commands refer to the command reference earlier in this chapter.

Creating a Report File

The report file contains the following information:

```
                                         ┌ ASCII    ┐            ┌ ON  ┐
REPORT [TITLE] title string FORMAT       │ DATAFILE │ HEADINGS   └ OFF ┘
                                         └ BINARY   ┘
SEPARATOR=char datatype items
```

where:

`REPORT TITLE`	(*Prints an optional character string and headings.*) The following page has more details.
`FORMAT`	(*Selects the data formats.*)
`ASCII`	An ASCII format file is similar to the HP LaserRX/MPE analysis software export to DATAFILE format. Files in ASCII format are the best files for copying to a printer or terminal.
`DATAFILE`	An ASCII-format file in which all non-numerical fields are enclosed in double quotes. Since double quotation marks makes it impossible maintain column alignment, files in DATAFILE format are not suitable for printing directly. DATAFILE format is the easiest format to import into most PC spreadsheets and graphics packages.
`BINARY`	Binary format is a more compact format that represents numerical values as binary integers. It is the most suitable format for input into user-written analysis programs. Since it requires the least amount of conversion, it maintains the highest metric accuracy, but it is not suitable for printing directly.

Within Syntax Statements

`HEADINGS`	(*Selects whether or not to include column headings in the data file (HEADINGS=ON or HEADINGS=OFF).*)
`SEPARATOR`	(*Selects the character that is printed between each field in the DATAFILE format.*) The default separator is a blank space, but many programs prefer the field separator to be a comma. You may set the separator to any printing or nonprinting character.
`DATA TYPE`	(*Selects one of the exportable data types: GLOBAL, APPLICATION, PROCESS, DISCSPACE, or CONFIGURATION.*) This starts a section of the report file that lists the data items to be copied when this type of data is exported.
`ITEMS`	(*Specifies the data to be included in the exported file.*) Item names are listed, one per line, in the order you want them listed in the resulting file. You must select the proper DATA TYPE before listing any ITEMS. You may include item lists for as many data types as you wish in the same report file. Each data type will be referenced *only if* you choose to export that type of data. See the tables later in this chapter for the data items for each data type.

You may have more than one report file on your system. Each one can define a set of exported file formats to suit a particular user's needs. You specify the report file to be used with the REPORT command when you run the EXTRACT program.

The following items may be substituted in the REPORT TITLE string:

!DATE	(*Date the EXPORT function was performed.*)
!TIME	(*Time the EXPORT function was performed.*)
!LOGFILE	(*The fully qualified name of the source performance log file.*)
!COLLECTOR	(*Name and version of the Performance Collector program.*)
!SYSTEM_ID	(*Identifier of the system that collected the data.*)

For example, the string

```
REPORT "Export !SYSTEM_ID data from !LOGFILE on !DATE !TIME"
```

would generate a report title similar to

```
Export COOKIE data from LOGGLOB.SCOPE.SYS on 02/02/91 08:30 AM
```

An Example of Exporting Data

Example: You want to export GLOBAL and APPLICATION data at a rate of one data point per hour for use in creating a custom graph or report. Take the following steps:

1. First, you must determine what data items you will need from each data type and in what format should you access them. For this example, assume we will be graphing Global Queue Depths and Application Response Times, you would like an ASCII file without headings, and each field will be separated by commas.
2. Create and save the following ASCII file. Call it REPORT1.

```
REPORT "Sample Report File (REPORT1)"
FORMAT ASCII
HEADINGS OFF
SEPARATOR=","

DATA TYPE GLOBAL

   CPUQUEUE
   DISCQUEUE
   MEMORYQUEUE
   IMPEDEQUEUE

DATA TYPE APPLICATION

   APPLICATION
   TRANSACTIONS
   FIRSTRESP
   PROMPT
```

3. Run the EXTRACT program.

```
:RUN EXTRACT
Enter command (or press softkey)
```

4. Select the report file generated.

```
REPORT REPORT1
```

5. Select GLOBAL SUMMARY data and APPLICATION SUMMARY data using standard EXTRACT program commands.

```
GLOBAL SUMMARY
APPLICATION SUMMARY
```

6. Now enter `EXPORT`—it means “GO”.

```
EXPORT
ENTER THE LOG FILE NAME (LOGGLOB.SCOPE.SYS)
```

7. Since you didn't tell the program from where it should get the performance data, the program will prompt you. In this example the default log file is correct, just press RETURN.

```
EXPORTING GLOBAL DATA .........50%......100%
EXPORTING APPLICATION DATA ....50%......100%

The exported file contains 31 days of data from 09/01/91 to 10/01/91

                          Examined  Exported
Data Type                  Records   Records     Space
-----------------------  --------- --------- ---------
GLOBAL                        8817         0   0.00 MB
GLOBAL      SUMMARIES                    736   0.20 MB
APPLICATION                  28491         0   0.00 MB
APPLICATION SUMMARIES                   2560   0.71 MB
                                             ---------
                                               0.91 MB
```

You are finished. You have just created two files—XFERGSUM and XFERASUM—that contain the global and application summary data in the format you specified.

Data Items for Exporting Data

The following data items are available for Data Type GLOBAL:

Table 5-2. EXPORT Items for Data Type GLOBAL

Record Identification Metrics	
`RECORD_TYPE`	ASCII field to identify this record type "GLOB".
`DATE`	Date in MM/DD/YY format (or custom NLS date).
`TIME`	Time in HH:MM 24-hour format.
`DAY`	Julian day-of-the-year (1-366).
`YEAR`	Year (such as 1991).
`DATE_SECONDS`	Date in UN*:X format (in seconds, since January 1, 1970).
`INTERVAL`	Time included in this sample (in seconds).
`SAMPLES`	Number of individual data samples averaged in this data.
`NUMBER_OF_DISCS`	Number of disc drives configured on the system.
`BLANK`	An empty field used as a spreadsheet place holder.
Summary Metrics	
`CPU_TOTAL`	Average overall CPU usage during the interval (percentage of total).
`CPU_SECONDS`	Average overall CPU usage during the interval (time, in seconds).
`DISC_TOTAL`	Average overall physical disc IO rate (IOs/second).
`DISC_IO`	Average overall physical disc IO count.
`CPU_HISTOGRAM`	A 60-character-wide histogram of CPU components.
`DISC_HISTOGRAM`	A 60-character-wide histogram of DISC IO components.

Table 5-2. EXPORT Items for Data Type GLOBAL (continued)

CPU Metrics	
CPU_SESSION	CPU usage by interactive sessions (percentage of total).
CPU_SESSION_SECONDS	CPU usage by interactive sessions (time, in seconds).
CPU_JOB	CPU usage by batch jobs (percentage of total).
CPU_JOB_SECONDS	CPU usage by batch jobs (time, in seconds).
CPU_SYSTEM	CPU usage by system processes (percentage of total).
CPU_SYSTEM_SECONDS	CPU usage by system processes (time, in seconds).
CPU_PAUSED	Time CPU was idle and Disc IO was occurring (percentage of total).
CPU_PAUSED_SECONDS	Time CPU was idle and Disc IO was occurring (in seconds).
CPU_CACHE	CPU usage by Disc Caching (percentage of total; MPE V only).
CPU_CACHE_SECONDS	CPU usage by Disc Caching (time, in seconds; MPE V only).
CPU_DISPATCH	CPU usage by the Dispatcher (percentage of total; MPE/iX only).
CPU_DISPATCH_SECONDS	CPU usage by the Dispatcher (time, in seconds; MPE/iX only).
CPU_ICS	CPU usage for other activities such as interrupts (percentage of total).
CPU_ICS_SECONDS	CPU usage for other activities such as interrupts (time, in seconds).
CPU_MEMMGR	CPU usage for memory management (percentage of total).
CPU_MEMMGR_SECONDS	CPU usage for memory management (time, in seconds).
CPU_IDLE	CPU usage when processor was not busy or paused (percentage of total).
CPU_IDLE_SECONDS	CPU usage when processor was not busy or paused (time, in seconds).
PROCESSOR1_BUSY	Percentage busy on first system processor.
PROCESSOR1_SECONDS	Time busy on first system processor (in seconds).
PROCESSOR2_BUSY	Percentage busy on second system processor (MPE/iX MP only).
PROCESSOR2_SECONDS	Time busy on second system processor (in seconds, MPE/iX MP only).
PROCESSOR3_BUSY	Percentage busy on third system processor (MPE/iX MP only).
PROCESSOR3_SECONDS	Time busy on third system processor (in seconds, MPE/iX MP only).
PROCESSOR4_BUSY	Percentage busy on fourth system processor (MPE/iX MP only).
PROCESSOR4_SECONDS	Time busy on fourth system processor (in seconds, MPE/iX MP only).

Table 5-2. EXPORT Items for Data Type GLOBAL (continued)

Disc Metrics	
DISC_LOGICAL	Logical disc IO rate (IOs/second).
DISC_LOGICAL_IO	Number of logical disc IO transfers (kilobytes transferred).
DISC_SESSION	Physical disc IO rate by interactive sessions (IOs/second).
DISC_SESSION_IO	Number of physical disc transfers by interactive sessions (kilobytes transferred).
DISC_JOB	Physical disc IO rate by batch jobs (IOs/second).
DISC_JOB_IO	Number of physical disc IO transfers by batch jobs (kilobytes transferred).
DISC_SYSTEM	Physical disc IO rate by system processes (IOs/second).
DISC_SYSTEM_IO	Number of physical disc transfers by system processes (kilobytes transferred).
DISC_MEMMGR	Physical disc IO rate for memory management (IOs/second).
DISC_MEMMGR_IO	Number of physical disc transfers for memory management (kilobytes transferred).
DISC_LOGLREAD	Logical disc read rate (IOs/second).
DISC_LOGLREAD_IO	Number of logical disc read transfers (kilobytes transferred).
DISC_LOGLWRITE	Logical disc write rate (IOs/second).
DISC_LOGLWRITE_IO	Number of logical disc write transfers (kilobytes transferred).
DISC_PHYSREAD	Physical disc read rate for user files (reads/second).
DISC_PHYSREAD_IO	Number of physical disc reads for user files.
DISC_PHYSWRITE	Physical disc write rate for user files (writes/second).
DISC_PHYSWRITE_IO	Number of physical disc writes for user files.
DISC_MEMREAD	Physical disc read rate for memory management (reads/second).
DISC_MEMREAD_IO	Number of physical disc reads for memory management.
DISC_MEMWRITE	Physical disc write rate for memory management (writes/second).
DISC_MEMWRITE_IO	Number of physical disc writes for memory management.
DISC_UTILIZATION	Peak disc utilization (busiest disc drive; percentage of total).
DISC_UTIL_SECONDS	Peak disc utilization (time busy on busiest drive, in seconds).
DISC_KBYTE/SECOND	Physical disc transfer rate (kilobytes per second).
DISC_KBYTE_COUNT	Number of physical disc transfers (kilobytes transferred).

Table 5-2. EXPORT Items for Data Type GLOBAL (continued)

Process Queue Depths (Load Factors)	
`CPUQUEUE`	Average number of processes waiting for or using CPU.
`DISCQUEUE`	Average number of processes waiting for DISC transfers.
`MEMORYQUEUE`	Average number of processes waiting for MEMORY.
`IMPEDEQUEUE`	Average number of processes waiting for IMPEDES (locks).
Job/Session Count Metrics	
`NUM_JOBS`	Average number of logged-on batch jobs.
`NUM_SESSIONS`	Average number of logged-on interactive sessions.
`ACTIVE_JOBS`	Average number of batch jobs using CPU resources.
`ACTIVE_SESSIONS`	Average number of interactive sessions using CPU resources.
`JOBS_COMPLETED`	Number of batch jobs that completed during the interval.
`JOBS_RUNTIME`	Average run time for completed batch jobs (in seconds).
`JOB_PROC_COMPLETED`	Number of batch processes that completed during the interval.
`JOB_PROC_RUNTIME`	Average run time for completed batch processes (in seconds).
`SESSIONS_COMPLETED`	Number of sessions that completed during the interval.
`SESSIONS_RUNTIME`	Average run time for completed sessions (in seconds).
`SESS_PROC_COMPLETED`	Number of session processes that completed during the interval.
`SESS_PROC_RUNTIME`	Average run time for completed session processes (in seconds).
Terminal Transaction Metrics	
`TRANSACTIONS`	Number of completed terminal transactions during the interval.
`TRANSACT/MIN`	Terminal transaction rate (transactions per minute).
`TRANSACT/HOUR`	Terminal transaction rate (transactions per hour).
`THINKTIME`	Average think time for terminal transactions (in seconds).
`FIRSTRESP`	Average first-response time for transactions (in seconds).
`PROMPT`	Average response-to-prompt time for transactions (in seconds).

Table 5-2. EXPORT Items for Data Type GLOBAL (continued)

Terminal Transaction Distributions. Each metric is an array of 10 items.

DISTRIBUTION_FIRST	First-response time distribution (10 values).
DISTRIBUTION_PROMPT	Response-to-prompt distribution (10 values).
DISTRIBUTION_THINK	Transaction think-time distribution (10 values).

Individual Disc Metrics. The following fields will be repeated once for every disc on the system.

@DISC_LOGLREAD	Logical read rate (IOs/second).
@DISC_LOGLREAD_IO	Number of logical reads.
@DISC_LOGLWRITE	Logical write rate (IOs/second).
@DISC_LOGLWRITE_IO	Number of logical writes.
@DISC_PHYSREAD	Physical read rate (IOs/second).
@DISC_PHYSREAD_IO	Number of physical reads.
@DISC_PHYSWRITE	Physical write rate (IOs/second).
@DISC_PHYSWRITE_IO	Number of physical writes.
@DISC_MEMREAD	Memory management read rate (IOs/second).
@DISC_MEMREAD_IO	Number of memory management reads.
@DISC_MEMWRITE	Memory management write rate (IOs/second).
@DISC_MEMWRITE_IO	Number of memory management writes.
@DISC_UTILIZATION	Percentage disc utilization.
@DISC_KBYTE/SEC	Physical disc transfer rate (kilobytes per second).
@DISC_KBYTE_COUNT	Number of physical disc transfers (kilobytes transferred).

Within Syntax Statements

APPLICATION Data Type

Application data type can generate two different record layouts. The first layout writes one record for every application that was active during the time interval. Each application is in a separate record.

The second layout writes only one record for each interval. In this layout, a section of the record is reserved for every application. If the application was not active during the interval then its fields will contain zeros.

You determine which application layout you will use by the metrics you choose. The application metrics will be listed in three sections. The first section ("Record Identification Metrics") can be used in either layout. Other application metrics may be selected from only one of the next two sections ("Single Application" and "Multiple Application" layouts, respectively).

Used with either application layout:

Table 5-3.
EXPORT Items for Data Type APPLICATION (Common)

Record Identification Metrics	
`RECORD_TYPE`	ASCII field to identify this record type "APPL".
`DATE`	Date in MM/DD/YY format (or Custom NLS date).
`TIME`	Time in HH:MM 24-hour format.
`DAY`	Julian day-of-the year (1-366).
`YEAR`	Year (such as 1991).
`DATE_SECONDS`	Date in UN*X format (in seconds, since January 1, 1970).
`NUMBER_OF_APPLS`	Number of applications defined.
`NUMBER_OF_DISCS`	Number of disc drives configured on the system.

Table 5-4. EXPORT Items for Data Type APPLICATION (Single Layout)

Used with single application layouts:

Application ID Metrics	
INTERVAL	Time covered by this record (in seconds).
SAMPLES	Number of samples averaged into this record.
APPLICATION_NO	Sequential application number (matches PARM file).
APPLICATION	Application name (20-byte ASCII).
BLANK	An empty field used as a spreadsheet place holder.
Summary Metrics	
CPU_TOTAL	CPU usage by the application during the interval (percentage of total).
CPU_SECONDS	CPU time used by the application during the interval (in seconds).
DISC_TOTAL	Physical disc IO rate (IOs/second).
DISC_IO	Number of physical disc IOs.
CPU Metrics	
CPU_LINEAR	CPU usage while in the linear execution queue (percentage of total).
CPU_LINEAR_SECONDS	CPU usage while in the linear execution queue (time, in seconds).
CPU_CS	CPU usage while in the "CS" execution queue (percentage of total).
CPU_CS_SECONDS	CPU usage while in the "CS" execution queue (time, in seconds).
CPU_DS	CPU usage while in the "DS" execution queue (percentage of total).
CPU_DS_SECONDS	CPU usage while in the "DS" execution queue (time, in seconds).
CPU_ES	CPU usage while in the "ES" execution queue (percentage of total).
CPU_ES_SECONDS	CPU usage while in the "ES" execution queue (time, in seconds).

Table 5-4.
EXPORT Items for Data Type APPLICATION (Single Layout)
(continued)

Disc Metrics	
`DISC_LINEAR`	Physical disc IO rate while in linear execution queue (IOs/second).
`DISC_LINEAR_IO`	Number of physical disc IOs while in the linear execution queue.
`DISC_CS`	Physical disc IO rate while in the “CS” execution queue (Ios/second).
`DISC_CS_IO`	Number of physical disc IOs while in the “CS” execution queue.
`DISC_DS`	Physical Disc IO rate while in the “DS” execution queue (IOs/second).
`DISC_DS_IO`	Number of physical disc IOs while in the “DS” execution queue.
`DISC_ES`	Physical disc IO rate while in the “ES” execution queue (IOs/second).
`DISC_ES_IO`	Number of physical disc IOs while in the “ES” execution queue.
Process Queue Depths	
`CPUQUEUE`	Average number of processes waiting for or using CPU.
`DISCQUEUE`	Average number of processes waiting for DISC.
`MEMORYQUEUE`	Average number of processes waiting for MEMORY.
`IMPEDEQUEUE`	Average number of processes waiting for IMPEDES (locks).
Process Count Metrics	
`AVE_PROCESSES`	Average number of processes in the application.
`ACTIVE_PROCESSES`	Average number of processes that used CPU.
`PROCESSES_COMPLETED`	Number of application processes that completed.
`PROCESSES_RUNTIME`	Average run time of completed processes (in seconds).
Terminal Transaction Metrics	
`TRANSACTIONS`	Number/Rate of terminal transactions completed during the interval.
`TRANSACTION_COUNT`	Number of terminal transactions completed during the interval.
`THINKTIME`	Average transaction think time (in seconds).
`FIRSTRESP`	Average transaction time-to-first-response (in seconds).
`PROMPT`	Average transaction response-to-prompt time (in seconds).
`SERVICE_LEVEL`	Percentage of transactions that met service level.
`SERVICE_LEVEL_COUNT`	Number of transactions that met service level.

Table 5-4. EXPORT Items for Data Type APPLICATION (Single Layout) (continued)

Average process Wait State Metrics

`STOPCPU`	Time processes spent waiting for CPU (percentage of total).
`STOPDISC`	Time processes spent waiting for DISC (percentage of total).
`STOPSWAP`	Time processes spent waiting for MEMORY (percentage of total).
`STOPIMPEDE`	Time processes spent IMPEDED (locked) (percentage of total).
`STOPOTHERIO`	Time processes spent waiting for other IO (percentage of total).
`STOPTERM`	Time processes spent waiting for terminal input (percentage of total).

Individual Disc Metrics. The following fields will be repeated once for each disc on the system:

`@DISC_LOGLREAD`	Logical read rate (IOs/second).
`@DISC_LOGLREAD_IO`	Number of logical reads.
`@DISC_LOGLWRITE`	Logical write rate (IOs/second).
`@DISC_LOGLWRITE_IO`	Number of logical writes.
`@DISC_PHYSREAD`	Physical read rate (IOs/second).
`@DISC_PHYSREAD_IO`	Number of physical reads.
`@DISC_PHYSWRITE`	Physical write rate (IOs/second).
`@DISC_PHYSWRITE_IO`	Number of physical writes.

Table 5-5. EXPORT Items for Data Type APPLICATION (Multiple Layout)

Used with multiple application layouts. The following fields will be repeated once for each application defined on the system.

Application Identification Metrics	
`@INTERVAL`	Time covered by this record (in seconds).
`@SAMPLES`	Number of samples averaged into this record.
`@APPLICATION_NO`	Sequential application number (matches PARM file).
`@APPLICATION`	Application name (20-byte ASCII).
`@BLANK`	An empty field used as a spreadsheet place holder.
Summary Metrics	
`@CPU_TOTAL`	CPU usage by the application during the interval (percentage of total).
`@CPU_SECONDS`	CPU usage by the application during the interval (time, in seconds).
`@DISC_TOTAL`	Physical disc IO rate (IOs/second).
`@DISC_IO`	Number of physical disc IOs.
CPU Metrics	
`@CPU_LINEAR`	CPU usage while in the linear execution queue (percentage of total).
`@CPU_LINEAR_SECONDS`	CPU usage while in the linear execution queue (time, in seconds).
`@CPU_CS`	CPU usage while in the “CS” execution queue (percentage of total).
`@CPU_CS_SECONDS`	CPU usage while in the “CS” execution queue (time, in seconds).
`@CPU_DS`	CPU usage while in the “DS” execution queue (percentage of total).
`@CPU_DS_SECONDS`	CPU usage while in the “DS” execution queue (time, in seconds)
`@CPU_ES`	CPU usage while in the “ES” execution queue (percentage of total).
`@CPU_ES_SECONDS`	CPU usage while in the “ES” execution queue (time, in seconds).
Disc Metrics	
`@DISC_LINEAR`	Physical disc IO rate while in linear execution queue (IOs/second).
`@DISC_LINEAR_IO`	Number of physical disc IOs while in the linear execution queue.
`@DISC_CS`	Physical disc IO rate while in the “CS” execution queue (IOs/second).
`@DISC_CS_IO`	Number of physical disc IOs while in the “CS” execution queue.
`@DISC_DS`	Physical disc IO rate while in the “DS” execution queue (IOs/second).

Table 5-5. EXPORT Items for Data Type APPLICATION (Multiple Layout) (continued)

@DISC_DS_IO	Number of physical disc IOs while in the "DS" execution queue.
@DISC_ES	Physical disc IO rate while in the "ES" execution queue (IOs/second).
@DISC_ES_IO	Number of physical disc IOs while in the "ES" execution queue.
Process Queue Depths	
@CPUQUEUE	Average number of processes waiting for or using CPU.
@DISCQUEUE	Average number of processes waiting for DISC.
@MEMORYQUEUE	Average number of processes waiting for MEMORY.
@IMPEDEQUEUE	Average number of processes waiting for IMPEDES (locks).
Process Count Metrics	
@AVE_PROCESSES	Average number of processes in the application.
@ACTIVE_PROCESSES	Average number of processes that used CPU.
@PROCESSES_COMPLETED	Number of application processes that completed.
@PROCESSES_RUNTIME	Average run time of completed processes (in seconds).
Terminal Transaction Metrics	
@TRANSACTIONS	Number or rate of terminal transactions completed during the interval.
@TRANSACTION_COUNT	Number of terminal transactions completed during the interval.
@THINKTIME	Average transaction think time (in seconds).
@FIRSTRESP	Average transaction time-to-first-response (in seconds).
@PROMPT	Average transaction response-to-prompt time (in seconds).
@SERVICE_LEVEL	Transactions that met service level (percentage of total).
@SERVICE_LEVEL_COUNT	Number of transactions that met service level.
Average Process Wait State Metrics	
@STOPCPU	Time processes spent waiting for CPU (percentage of total).
@STOPDISC	Time processes spent waiting for DISC (percentage of total).
@STOPIMPEDE	Time processes spent IMPEDED (locked) (percentage of total).
@STOPOTHERIO	Time processes spent waiting for other IO (percentage of total).
@STOPSWAP	Time processes spent waiting for MEMORY (percentage of total).
@STOPTERM	Time processes spent waiting for terminal input (percentage of total).

Within Syntax Statements

PROCESS Data Type

The following data items are available for the PROCESS data type. There is one record for each interesting process.

Table 5-6. EXPORT Items for Data Type PROCESS

Record Identification Metrics	
RECORD_TYPE	ASCII field to identify this record type "PROC".
DATE	Date in MM/DD/YY format (or custom NLS date).
TIME	Time in HH:MM 24-hour format.
DAY	Julian day-of-the-year (1-366).
YEAR	Year (such as 1991).
DATE_SECONDS	Date in UN*X format (in seconds, since January 1, 1970).
INTERVAL	Time included in this sample (in seconds).
BLANK	An empty field used as a spreadsheet place holder.
Process Identification Metrics	
PIN	Process Identification Number.
PROGRAM	Program name (or ":" last MPE command).
JSNO	Job/Session number (J###/S### or "SYS").
LDEV	Logon device number.
LOGON	User logon string.
QUEUE	Execution queue (L,A,B,C,D,E).
PRIORITY	Execution priority at end of interval.
INTEREST	Codes for reason of interest (*see* table 5-7).
STOP_REASON	Reason process last stopped execution.
APPLICATION_NO	Sequential application number (matches PARM file).

Table 5-6. EXPORT Items for Data Type PROCESS (continued)

Summary Metrics	
`RUNTIME`	Total process execution time until end of interval.
`TOTAL_CPU`	Percentage of CPU usage during process life (percentage of total run time).
`TOTAL_CPU_SECONDS`	Time of CPU usage during process life (run time, in seconds).
`TOTAL_DISC`	Physical disc IO rate during process life (IOs/second).
`TOTAL_DISC_IO`	Number of physical disc IOs during process life.
`TOTAL_FIRSTRESP`	Average transaction first-response time during process life.
`TOTAL_PROMPT`	Average transaction response-to-prompt time during process life.
`TOTAL_THINK`	Average transaction think time during process life.
`TOTAL_TRANS`	Number or Rate of transactions during process life.
`TOTAL_TRANS_COUNT`	Number of transactions during process life.
CPU Metrics	
`CPU`	CPU usage during the interval (percentage of total).
`CPU_SECONDS`	CPU usage during the interval (time, in seconds).
`CM`	CPU usage that was in compatibility mode (percentage of total).
`CM_SECONDS`	CPU usage that was in compatibility mode (time, in seconds).
`SWITCHTOCM`	Rate of switches from native mode to compatibility mode (switches/second).
`SWITCHTOCM_COUNT`	Number of switches from native mode to compatibility mode.
`SWITCHTONM`	Rate of switches from compatibility mode to native mode (switches/second).
`SWITCHTONM_COUNT`	Number of switches from compatibility mode to native mode.
Disc metrics	
`DISC`	Physical disc IO rate (IOs/second).
`DISC_IO`	Number of physical disc IOs.
`LOGLREAD`	Logical disc read rate during interval (IOs/Second).
`LOGLREAD_IO`	Number of logical disc reads during the interval.
`LOGLWRITE`	Logical disc write rate during interval (IOs/Second).
`LOGLWRITE_IO`	Number of logical disc writes during the interval.

Table 5-6. EXPORT Items for Data Type PROCESS (continued)

PHYSREAD	Physical disc read rate during interval (IOs/Second).
PHYSREAD_IO	Number of physical disc reads during the interval.
PHYSWRITE	Physical disc write rate during interval (IOs/Second) .
PHYSWRITE_IO	Number of physical disc writes during the interval.
Terminal Transaction Metrics	
TRANSACTIONS	Number or rate of transactions completed during interval.
TRANSACTION_COUNT	Number of transactions completed during the interval.
THINKTIME	Average transaction think time (in seconds).
FIRSTRESP	Average transaction time-to-first response (in seconds).
PROMPT	Average transaction response-time-to-prompt (in seconds).
Process Wait States Metrics	
STOPCPU	Time process spent waiting for CPU (percentage of total).
STOPCPU_SECONDS	Time process spent waiting for CPU (in seconds).
STOPDISC	Time process spent waiting for DISC IOs (percentage of total).
STOPDISC_SECONDS	Time process spent waiting for DISC IOs (in seconds).
STOPIMPEDE	Time process spent IMPEDED (locked)(percentage of total).
STOPIMPEDE_SECONDS	Time process spent IMPEDED (locked, in seconds).
STOPOTHERIO	Time process spent waiting for OTHER IO (percentage of total).
STOPOTHERIO_SECONDS	Time process spent waiting for OTHER IO (in seconds).
STOPSWAP	Time process spent waiting for MEMORY (percentage of total).
STOPSWAP_SECONDS	Time process spent waiting for MEMORY (in seconds).
STOPTERM	Time process spent waiting for TERMINAL input (percentage of total).
STOPTERM_SECONDS	Time process spent waiting for TERMINAL input (in seconds).

Table 5-7. Codes for Reason of Interest

The INTEREST field consists of 12 independent columns. Each column contains either a blank or a character representing a process INTEREST code, as shown below.

Position	Character	Meaning
1	`N`	New (process is newly created).
2	`K`	Killed (process is terminated).
3	`C`	CPU percentage exceeded threshold.
4	`D`	Disc I/Os exceeded threshold.
5	`P`	Response-to-prompt exceeded threshold.
6	`F`	First-response exceeded threshold.
7	`T`	Transaction rate exceeded threshold.
8	`c`	Wait on CPU percent exceeded threshold.
9	`d`	Wait on disc percent exceeded threshold.
10	`m`	Wait on memory percent exceeded threshold.
11	`i`	Wait on impede percent exceeded threshold.
12	*blank*	Not used.

Within Syntax Statements

DISC Data Type

The following data items are available for the DISC data type.

Table 5-8. EXPORT Items for Data Type DISC SPACE

Record Identification Metrics	
`RECORD_TYPE`	ASCII field to identify this record type "DISC".
`DATE`	Date in MM/DD/YY format (or custom NLS date).
`TIME`	Time in HH:MM 24-hour format.
`DAY`	Julian day-of-the-year (1-366).
`YEAR`	Year (such as 1991).
`DATE_SECONDS`	Date in UN*X format (in seconds, since January 1, 1970).
`BLANK`	An empty field used as a spreadsheet place holder.
`NUMBER_OF_DISCS`	Number of discs configured on the system.
`NUMBER_OF_GROUPS`	Number of disc groups logged in this record.
Overall Disc Space Metrics	
`CAPACITY`	Total disc capacity on system (sectors).
`TOTAL_FREE`	Total free disc space in sectors (Virtual+Permanent).
`LARGEST_FREE`	Largest contiguous free space (sectors).
Permanent Files Disc Space Metrics	
`FILES`	Total disc space used by permanent files (sectors).
`FREE_PERMANENT`	Total disc space available for permanent file usage.

Table 5-8. EXPORT Items for Data Type DISC SPACE (continued)

Transient (Virtual) Disc Space Metrics[1]

`TRANSIENT_CAPACITY`	Total disc space reserved for transient or virtual memory.
`VIRTUAL_CAPACITY`*	Total disc space reserved for transient or virtual memory.
`PEAK_TRANSIENT`	Peak transient/virtual memory usage during the last day.
`PEAK_VIRTUAL`*	Peak transient/virtual memory usage during the last day.
`TRANSIENT`	Total disc space used by transient/virtual memory.
`VIRTUAL`*	Total disc space used by transient/virtual memory.
`FREE_TRANSIENT`	Total disc space available for use by transient objects.
`FREE_VIRTUAL`*	Total disc space available for use by virtual memory.

Free Disc Space Fragmentation Metrics

`FRAGMENT99`	Total disc space in fragments <=99 sectors in size.
`FRAGMENT1K`	Total disc space in fragments 100-999 sectors.
`FRAGMENT10K`	Total disc space in fragments 1,000-9,999 sectors.
`FRAGMENT100K`	Total disc space in fragments 10,000-99,999 sectors.
`FRAGMENT_OVER100K`	Total disc space in fragments 100,000 sectors and larger.

Permanent Disc Space User Metrics. The following fields will be repeated once for every disc group logged.

`@GROUP_NAME`	Name of the disc group (or account name).
`@GROUP_SECTORS`	Sectors used for permanent files.

[1] The terms TRANSIENT and VIRTUAL are used by MPE/iX and MPE V, respectively. To allow the same report files to be used on different types of systems without modification, the terms "TRANSIENT" and "VIRTUAL" may be used interchangeably. For Example: "PEAK_TRANSIENT" and "PEAK_VIRTUAL" represent the *same* metric and may be used *on either* system.

Within Syntax Statements

CONFIGURATION Data Type

The following data items are available for the CONFIGURATION data type. There is one record for each collector start-up or NEWPARM.

Table 5-9. EXPORT Items for Data Type CONFIGURATION

Record Identification Metrics	
`RECORD_TYPE`	ASCII field to identify this record type "CONF".
`DATE`	Date in MM/DD/YY format (or custom NLS date).
`TIME`	Time in HH:MM 24-hour format.
`DAY`	Julian day-of-the-year (1-366).
`YEAR`	Year (such as 1991).
`DATE_SECONDS`	Date in UN*X format (in seconds, since January 1, 1970).
`BLANK`	An empty field used as a spreadsheet place holder.
System Identification Metrics	
`SYSTEM_ID`	System identification string.
`SYSTEM_TYPE`	Type of system (3000/960, etc.).
`SERIAL_NO`	System hardware serial number (HPSUSAN, etc.).
`OP_SYS_NAME`	Operating system name (MPE V, MPE/iX, HP-UX etc.).
`OP_SYS_VERSION`	Operating system release number.
`MEMORYSIZE`	Size of main memory (kilobytes).
`USERMEMORY`	Amount of main memory available for nonresident programs.
`SWAPMEMORY`	Amount of swap memory available on disc (HP-UX only).
`NUMPROCESSORS`	Number of processors configured.
`NUMBER_OF_DISCS`	Number of disc drives configured.
`NUMBER_OF_DATACOMM`	Number of LAN interfaces (not implemented on MPE).
Collector Identification Metrics	
`COLLECTOR`	Name and version of the Performance Data Collection program.
`LOGFILE_VERSION`	Version of the log file (*A* or *B*).
`LOGGING_TYPES`	Types of performance data being logged (G, A, P, or D).

Table 5-9.
EXPORT Items for Data Type CONFIGURATION (continued)

Interesting Process Thresholds

`THRESHOLD_CPU`	Interesting process CPU threshold setting (percentage).
`THRESHOLD_DISC`	Interesting process DISC threshold setting (IOs/second).
`THRESHOLD_TRANS`	Interesting process TRANSACTION threshold (number).
`THRESHOLD_FIRSTRESP`	Interesting process FIRST RESPONSE threshold (in seconds).
`THRESHOLD_RESPONSE`	Interesting process RESPONSE threshold (in seconds).
`THRESHOLD_NOKILLED`	Interesting process `Don't log KILLED processes` flag.
`THRESHOLD_NONEW`	Interesting process `Don't log NEW processes` flag.
`THRESHOLD_NOSHORT`	Interesting process `Don't log SHORT processes` (in seconds).
`THRESHOLD_MINTHINK`	Transaction minimum think time threshold (in seconds).
`THRESHOLD_MAXTHINK`	Transaction maximum think time threshold (in seconds).
`WAIT_CPU`	Interesting process waiting for CPU threshold (percentage).
`WAIT_DISC`	Interesting process waiting for DISC threshold (percentage).
`WAIT_MEMORY`	Interesting process waiting for MEMORY threshold (percentage).
`WAIT_IMPEDE`	Interesting process waiting for IMPEDE threshold (percentage).

Terminal Transaction Distribution Bounds

`DISTRIBUTION_FIRST`	Lower limits of first-response distribution (10 values).
`DISTRIBUTION_PROMPT`	Lower limits of response-to-prompt distribution (10 values).
`DISTRIBUTION_THINK`	Lower limits of Think time Distribution (10 values).

Individual Disc Logical Device Numbers. The following field will be repeated once for every disc on the system.

`@DISC_LDEV`	Logical device number of disc drive.

Resulting Exported Files

By default, exported files are created with the following characteristics. You may use file equations to override these defaults, if you wish.

Table 5-10. Characteristics of Exported Files

Maximum Number Of Records:		50,000
Record Width:		Adjusted, based on items chosen
		Maximum: MPE/iX 8000 Bytes MPE V 4000 Bytes
Record Format:		Variable
Record Type:		ASCII (for ASCII and DATAFILE)
		BINARY (for BINARY)
File Name:	Global Data:	XFERGLOB
	Global Summary Data:	XFERGSUM
	Application Data:	XFERAPPL
	Application Summary:	XFERASUM
	Process Data:	XFERPROC
	Disc Space Data:	XFERDISC
	Configuration Data:	XFERCONF

The maximum number of discs supported in the repeating fields is:

MPE/iX:	64
MPE V:	32

Note Currently, the MPE/iX Application Records are recorded with only *one* detail disc drive. This drive contains the summary of all other drives.

The contents of each file will be:

```
REPORT TITLE LINE   (If REPORT Title was specified and HEADINGS=ON.)
HEADING LINE1       (If HEADINGS=ON.)
HEADING LINE2       (If HEADINGS=ON and it is not a binary file.)
FIRST  DATA RECORD
SECOND DATA RECORD
```

Report title and heading lines will *not* be repeated in the file.

FIXED versus VARIABLE Length Record Formats

By default, the exported files will be built with *variable-length* records. If a report format has repeating fields, only the valid number of fields will be written to the file. Using variable-length records can save considerable disc space. If your application can not handle variable-length records, you can override it with the following file equation:

```
:FILE XFERGLOB;REC=,,F
```

Each record will be padded to fill the maximum record size. The records are padded with blanks for ASCII and DATAFILES, and with binary zeros for BINARY files.

Maximum Record Width

Do not override the maximum record width for an exported file. If you try to write to a record that is too small, you might abort the EXTRACT program. If you want to truncate the records, export to the default variable-length record file, then copy the file before truncating it.

Details on ASCII and Data File Formats

All data in these format files should be in printable ASCII format. ASCII and DATAFILE formats are identical except that in the latter, all non-numeric fields are enclosed by double quotes. Even the DATAFILE HEADER information will be quoted. ASCII file format does not use double quotes to enclose fields, therefore these files will be more aligned when printed. DATAFILE format is best when used for PC analysis programs.

Within Syntax Statements

Numerical values will be formatted based on their range of values and internal accuracy. All fields will not be the same length so be sure to decide how you will determine the start of each field.

The user specified `SEPARATOR` character (or the default blank) will be used to separate each field from the next in ASCII and DATAFILE formats. Blank separators can be visually more attractive if you plan to print the report. Other separator characters may be more useful if you plan on reading the report file with another program. A commonly acceptable format for many PC applications is to use the comma as a separator.

```
SEPARATOR = ","
```

Be aware that some data items may contain commas that are not separators, and this can confuse the analysis programs. The process LOGON item will often contain a comma to delimit the job or session name and the logon group from the user and account names. The date and time formats may contain different special characters based on the Native Language chosen when the EXTRACT program is run.

Note To enter a nonprinting special character, enter it into your report format file following the first double quote. You might have to turn on the DISPLAY FUNCTIONS feature to enable your terminal to transmit the character to the file.

Hint Most spreadsheets will accept files in DATAFILE format using `SEPARATOR=" , "`.

On some packages, the horizontal tab character can be used to delimit columns. You can specify the horizontal tab character by typing `SEPARATOR= "`(TAB)`"`. On some terminals you might need to turn on the DISPLAY FUNCTIONS feature to actually enter the horizontal tab character properly.

Hint For a more attractive printout, try specifying ASCII format and the vertical bar character (`SEPARATOR="|"`), then print with underlining turned on, *if you have a printer that supports this.*

Details on Binary Format

Binary format files will write numerical values as 32-bit integers. This format can save space by reducing the overall file size, but it requires a program that is able to read binary files. Copying a binary-format file to a printer or a terminal is not recommended.

Non-numerical data will be written the same as it was in the ASCII format except separator characters will not be used. To properly utilize a *binary* format file, you should use the record layout report printed by EXTRACT when you specify `REPORT` *reportfile*`,SHOW`. This report will give you the starting byte of each item specified.

To maintain maximum precision and avoid nonstandard, binary floating-point representations, all numerical values will be written as scaled 32-bit integers. Some items may be multiplied by a constant before they are truncated into integer format.

For example, the number of seconds of CPU time used is multiplied by 1000 before being truncated. To convert the value in the exported file back to the number of seconds, divide it by 1000. The scale factors can be written to the exported file for ease in conversion by specifying `HEADERS=ON`. The report title, if specified, and a single header record will be written preceding the binary data.

Binary Header Record Layout

If the RECORD_TYPE item is selected, it will be written at the same location as it is in the data records. The ASCII code will be the same as in the data record, except it will be in lower case (glob, appl, proc, disc, conf). This can be useful because it insures that you are looking at a header record and not a report title or data record.

Within Syntax Statements

Consider the header record as an array of 32-bit integers. All data in the data records are aligned to a 32-bit boundary. If the data record contains a numerical value, then the header record will contain its scale factor in the corresponding location. Non-numeric items will have zeros in all corresponding locations in the header record. Negative numbers may be used to indicate an invalid item specification. (Invalid items should not be passed by the report file parser so negative scale factors indicate a program bug).

Special Scale Factors

The ASCII string, RECORD_TYPE, will appear as a very large scale factor if you try to decode it as an integer.

The scale factor for DATE is 512. DATE format is MPE CALENDAR format in the least significant 16 bits of the field—the rightmost 16 bits. Scaling this as a 32-bit integer (dividing by 512) will isolate the year as the integer part of the date, and the day of the year (divided by 512) as the fractional part.

The scale factor for TIME is 65536. TIME is a four-byte binary field (hour, minute, second, tenths of seconds). Dividing by 65536 will form a number where the integer part is the (hour $*$ 256) + MINUTE.

It may be easier to handle a DATE_SECONDS value in a binary file.

6

Archiving Strategies

Efficient analysis of system performance depends on how easily it is to access the performance data you collect. If disc space is not limited, you can keep several years of performance data online. Since usually you can analyze only the most recent data, you do not have to allocate much disc space to collected data.

Use the EXTRACT program to specify the type and amount of log file data you will need for later performance analysis.

The best procedure is to extract desired data periodically and summarize it to reduce disc space, if necessary. Release disc space by placing the data into one or more extracted files and transferring them to magnetic tape for offline storage.

Access the archived data by using RESTORE to bring the file online. You can either access the data directly, as a remote file, or EXTRACT the restored extracted file into another data subset before transferring it to your PC's local disc.

The EXPORT function *is not recommended* for archival use because Performance Collection Software programs cannot process data in export format.

This chapter discusses archiving techniques and a recommended archiving strategy.

Recommended Strategy

Extract detail data from raw log files into a different log file every month by using the MONTHLY command. Name each extracted file to permit identification and later restoration from archive tapes. If your system generates more than 30 megabytes of monthly data, you might need to extract data using the WEEKLY command or eliminate process detail data.

To save disc space, next extract only global and application summaries into a YEARLY file. Use this file for the HP RXForecast program and for examining long-term trends. The following batch job will perform these extractions:

```
:JOB MONTHLY,SCOPE.SYS,SCOPE
:RUN EXTRACT
GLOBAL BOTH
APPLICATION BOTH
PROCESS DETAIL
DISC DETAIL
MONTHLY
*****
GLOBAL SUM
APPLICATION SUM
PROCESS OFF
DISC DETAIL
YEARLY
EXIT
:EOJ
```

Use this job stream as follows:

- Stream the job periodically—at least monthly, preferably every few days.
- Whenever you have two files named RXMO####, you can STORE the file with the lowest number to tape and purge it from the disc. Timing is not critical since the raw log files act as a buffer as long as they contain all log file records collected between running the "monthly" jobs. The RXMO#### files will be ready for transferring data to tape storage as soon as the first monthly job is run.
- Once a year, you will get a second RXYR#### file. You can transfer the old RXYR#### file to tape and purge it from your disc.

Other Archiving Techniques

You can use Performance Collection Software log file data in several ways. Shortly after it is logged, you can use it to diagnose short-term problems on your system (or systems). To do this, you must be able to access the latest log file data quickly and in full detail.

If you have direct data communication links between your PC and the host system, the fastest way to access log file data is by opening a raw log file directly as a remote log file. This lets you see data that is no more than 5 minutes old.

If you do not have direct data communications between your PC and the host system, you can use the EXTRACT program to extract data from the raw log files for transmission to the PC or to a host that the PC can access directly. This extraction can be performed on demand either when needed or periodically in anticipation of a need.

For example, you may need to look at a host system when time is not critical. To handle questions once a week, you only have to examine the previous week's data for each system. In this case, you can schedule a weekly data extraction on each system and transfer the extracted log file to a central site—a PC or the host system.

Case 1

You want to have the previous week's data from each host system at a central site. This will help you minimize the demands of making extractions during prime time shift operations and to give you rapid access to data. Data from the previous week is satisfactory for analysis since it will not be affected by aging.

Technique

You want to retain each system's weekly log files. To make this manageable, set up each remote system in a different group on the central system. Run the remote job stream late Sunday night to take advantage of the WEEKLY extraction command.

Within Syntax Statements

A typical job stream is as follows:

```
:JOB SUNDAY,SCOPE.SYS,SCOPE
:RUN EXTRACT
GLOBAL BOTH
APPLICATION BOTH
PROCESS ON
DISC ON
WEEKLY
EXIT
:DSLINE CENTRAL
:REMOTE HELLO ARCHIVE.LOGFILES,BERT
:DSCOPY RXWEyyww TO ,CENTRAL;MOVE;REP
:REMOTE BYE
:EOJ
```

Now you can access any week on the BERT system by opening the appropriate log file. (The third week of 1991 for the BERT system is in a file called RXWE9103.BERT.ARCHIVE.)

Case 2

Case 2 is the same as case 1, except in case 2 you will overwrite each system's log files weekly.

Technique

Construct a batch job to run on each host system at the end of the week when prime time shift operations have finished. Have the job extract the data for the previous week and transmit it to the central site for access.

The new batch job would be:

```
:JOB SUNDAY,SCOPE.SYS,SCOPE
:RUN EXTRACT;PARM=7
GLOBAL BOTH
APPLICATION BOTH
PROCESS ON
DISC ON
OUTPUT RXBERT
EXTRACT
EXIT
:REMOTE HELLO ARCHIVE.LOGFILES;DSLINE=CENTRAL
:DSCOPY RXBERT TO ,CENTRAL;MOVE;REP
:REMOTE BYE
:EOJ
```

TIP You can save disc space and reduce transmission time at the expense of 5-minute data points on the Performance Collection Software graphs by specifying global summaries and application summaries to the EXTRACT program. You can save more disc space and transmission time by not extracting process data, although this data could be useful for diagnostic work.

At the central site, you can analyze this system by opening the remote log file RXBERT.ARCHIVE.LOGFILES, if central-site security allows the SCOPE.SYS user to read this file.

Case 3

Each host system must ensure that data is not lost even though the raw log files are busy during the system's daily backups. This allows SCOPE(XL) to run continuously and be available to analyze any performance problems, even those occurring during backup.

Within Syntax Statements

Technique

Perform a monthly extraction every day immediately before backup. (The MONTHLY extract command creates a single log file for each month, but appends new data to that log file whenever the command is executed.) Perform the following commands immediately before system backup:

```
:RUN EXTRACT
GLOBAL DETAIL
APPLICATION DETAIL
PROCESS ON
DISC ON
MONTHLY
EXIT
```

Remember that the extracted files are generated in the LOGON group.

The file created—RXMO*yymm*, where *yymm* is the year and month—is free to be written to the backup tape. Each month a new file is created with a higher numerical designation in its name. After a new monthly log file is created, the previous month's log file can be transferred to tape and the disk purged to recover disc space. If the stored data is ever needed, the file can be restored, and data can be extracted from it using the EXTRACT program.

Case 4

You want to determine long-term trends in a system's data. This allows you to track application CPU utilization over periods of a year or longer, or to make effective use of a forecasting tool such as HP RXForecast. In this case, you will not need detailed data, and the diagnostic nature of the process data is not worth the disc space it would require.

Technique

Save global and application summary data and disc detail data since it requires little additional disc space and it will be available for later examination. Stream the job at least once a month (more often if the size of your raw log files will hold less than two months of raw global and application data).

The following is a typical job stream:

```
:JOB SUMMARY,SCOPE.SYS,SCOPE
:RUN EXTRACT
GLOBAL SUMMARY
APPLICATION SUMMARY
PROCESS OFF
DISC ON
YEARLY
EXIT
:EOJ
```

You can access the remote log file RXYR*yyyy* (where *yyyy* is the year desired) or transfer this file to your PC for local access. Usually, the level of summarization you select will store a year of data in about 5 megabytes of disc space, or less (consistent with the program's other space requirements).

Log File Resizing Strategies

When raw log files are filled with data, they are *rolled back* automatically by the SCOPE(XL) data collection program. When a log file is rolled back, a new log file the same size as the old log file is built, the latest 75 percent of the data from the old file is copied to the new file, the old file is purged, then the new file is renamed with the old file's name. The net effect is to discard the oldest 25 percent of the data to make room for new data.

Case 1

The simplest strategy for managing raw log files is to do nothing and let the automatic process take care of the logs. You might want to select the amount of data each log file can hold to be sure that you have a specific amount of raw data available at all times. After logging data for several days, you can use the UTILITY program's RESIZE command to size each raw log file. Choose a size that will provide the desired amount of data when the log file is 75 percent full without using excessive disc space.

For example, if you want to have 30 days of global data available at all times to allow you to prepare one-month global graphs, size the global log file to hold 40 days of data (75 percent of 40 equals 30).

Hint Especially on MPE V host systems, avoid sizing raw log files too large, since this can slow access to the data in those files. MPE/iX host systems are not as sensitive to the size of raw log files.

Case 2

Since rolling back a log file can use significant system resources, you may want to schedule when it will occur. If you fill a log file during the prime-time shift, when maximum system usage occurs, the roll-back operation might compete with other processes for existing resources. This can cause online response times to degrade for as long as 5 minutes on heavily-loaded systems with large log files. Also, you may miss logging important performance data because data collection is suspended during the log file roll back operation.

You can instruct the SCOPE(XL) collector to resize log files at scheduled times by adding the MAINTTIME directive to the PARM file. The SCOPE(XL) program will resize a log file at the specified time if the program expects the file to become full within the next 24 hours. Since log records may accumulate at different rates depending on your system's activity, scheduled maintenance may resize log files too early or too late. In such cases, you might want to control log-file resizing more closely.

You can avoid unscheduled log file roll backs by using the UTILITY program to ensure that the raw log files contain sufficient empty space. Use the UTILITY RESIZE command to perform a scheduled log-file roll back. For example, once a week you can run a batch job that ensures there is enough room in the raw log files to hold another week of data. If there is not enough empty space, the log file will be rolled back, and old data will be discarded to provide the desired space. For example,

```
:JOB MAKEROOM,SCOPE.SYS,SCOPE
:RUN UTILITY
SCOPE KILL      (To turn the data collection program off.)
LOG LOGGLOB
SCAN            (To determine how much data is logged daily.)
RESIZE GLOBAL EMPTY=10 MAYBE
RESIZE APPLICATION EMPTY=10 MAYBE
RESIZE PROCESS EMPTY=10 MAYBE
RESIZE DISCSPACE EMPTY=10 MAYBE
EXIT
:STREAM SCOPEJOB (To restart the data collection program.)
:EOJ
```

The MAYBE parameter is added to the RESIZE command to force resizing *only* if the empty space currently in the log file is less than the requested amount. Ten days is chosen instead of seven days in case the amount of data logged each day increases slightly.

Because this operation might involve discarding data from raw log files to make room for new data, you should archive existing data *before* proceeding with this action.

Combined Recommendations

On each system, extract MONTHLY DETAIL and YEARLY SUMMARY data by using the EXTRACT program's MONTHLY and YEARLY commands. If you must access a system's data that is *not* directly accessible by your PC, perform an additional WEEKLY or DAILY extraction and move that data to a central analysis site where the PC can access it. If the PC can access a system directly, *do not* extract the daily and weekly data—accessing the raw log files directly as remote files usually works better.

Here is a combined job stream that performs all extractions. Execute it immediately before the daily or weekly backup depending on how much log file data you are willing to lose versus the amount of processing time the extraction will take. This is determined empirically for each system.

```
:JOB EXTRACT,SCOPE.SYS,SCOPE
:RUN EXTRACT
************************************************************
* First, top off the yearly summary data.
*
GLOBAL SUMMARY
APPLICATION SUMMARY
PROCESS OFF
DISC ON
YEARLY
************************************************************

* Next, top off the monthly detail data.
*
GLOBAL BOTH
APPLICATION BOTH
PROCESS ON      (or PROCESS OFF to save disc space)
DISC ON
MONTHLY
***********************************************************
```

```
* Finally, create the weekly extraction data.
* [FOR REMOTE SYSTEMS ONLY]
*
PROCESS ON     (Just in case we turned it off above.)
WEEKLY
************************************************************
EXIT
:COMMENT ***************************************************
:COMMENT FOR REMOTE SYSTEMS, MOVE THE WEEKLY DETAIL DATA
:COMMENT TO THE CENTRAL SYSTEM.
:COMMENT
:REMOTE HELLO THISSYS,REMOTE.LOGFILES,THISSYS;DSLINE=CENTRAL
:DSCOPY RXWE#### TO ,CENTRAL;REP;MOVE
:REMOTE BYE
:COMMENT ***************************************************
:TELLOP HP LaserRX EXTRACT JOB IS NOW FINISHED
:EOJ
```

Check the files once a month. Store and then purge any files named RXMO*yymm* when the next month's log file shows up. You could also store and purge any RXYR*yyyy* files once a year although it will not be necessary since they use such a small amount of disc space compared to the volume of data they contain.

There are also other archival strategies you can use. The EXTRACT program is flexible enough to meet your needs.

7 Performance Collection Software Performance Alarms

Performance Collection Software performance alarms may be used to identify periods in the logged data when performance met user-specified criteria. You can specify up to 50 independent alarms. Each alarm definition may contain the following:

- An alarm identifier (up to 40 characters).
- One or more CONDITIONS that must be satisfied.
- An ACTION to be performed when the alarm is initiated.
- An ACTION to be performed when the alarm has ended.
- A minimum time between repetitions of the same alarm.

The alarm identifier determines which alarm is being acted upon. The 40-character ID string is optional and may be any user-specified value.

A condition specifies two items, a comparison between those items, *and* a duration for which the comparison must be true. Items can be selected from a list of performance metrics provided by the SCOPE(XL) collector, constants, or user-defined variables. A user can define up to 26 user variables which are combinations of two items (metric, constant, or previously-defined user variables). These two items may be added, subtracted, multiplied, or divided to arrive at the new user-variable value.

You can specify actions to take when (1) the alarm is first satisfied because all conditions for the alarm were true for their specified durations *and/or* at the end of an alarm because one or more of the conditions ceases to be true *after* the alarm was satisfied. An action can be an MPE command (including :RUN commands), a command to the SCOPE(XL) collector, or the name of an ASCII file that may itself contain commands. The commands can include codes to force the substitution of metric or user variables before they are executed.

Within Syntax Statements

You can specify a minimum time between repetitions to prevent alarms from acting more than once within a given period. If you specify an alarm action such as the streaming of a detailed data collection job, you might want not want the job to be streamed more then once a day, even if the alarm condition continues through the day or is satisfied then becomes active again during the same day.

Note Alarm actions will not be performed while logged data is being examined. They can be listed with substituted performance metrics if desired.

Performance Collection Software alarms are specified in the SCOPE(XL) collector's PARM file. The syntax for specifying these alarms is:

```
ALARM=alarmid [TYPE=typeid] [SEVERITY=severitynumber]

IF item1 [>, <, >=, <= ]item2 FOR duration [MINUTES]

     [:MPECOMMAND          ]
THEN [$SCOPE scopecommand  ]
     [^COMMANDFILENAME     ]

       [:MPECOMMAND          ]
FINISH [$SCOPE scopecommand  ]
       [^COMMANDFILENAME     ]

REPEAT=nnn

VAR [A - Z]= item1 [+, -, *, / ]item2
```

where:

ALARM=

alarmid is a string of up to 40 characters that identifies this alarm. The default is all blanks.

typeid is a string of up to 8 characters that identifies the general category for this alarm. Any combination of 8 characters is acceptable, but certain products might recognize specific strings. The default is all blanks.

severity number is an integer number indicating the relative severity of this alarm. Any value between 0 and 32767 is acceptable although certain products might place special significance on values in certain ranges. The default is zero.

IF

item1 is the first item in the comparison. An item can be one of the following:

- A positive number specifying a literal constant [real numbers are allowed].
- A metric name chosen from the Performance Items tables shown later in this chapter.
- The upper case characters **A** through **Z** signifying a previously-defined user variable. VAR followed by upper case A though Z may also be used to reference a user variable. For example, `A` and `VAR A` are equivalent.

condition is one of the following codes that indicate how to compare item1 and item2.

>	true if item1 is greater than item2
<	true if item1 is less than item2
>=	true if item1 is greater than or equal to item2
<=	true if item1 is less than or equal to item2

duration is the number of continuous minutes that the condition must be true before the alarm is satisfied. Performance metrics are updated every five minutes and most are averages of the values for the last five-minutes. If at the end of a sample, the condition is true then five minutes is added to the accumulated time.

If the condition is false for more than one-half the duration time, the accumulated true time is reset to zero. The condition is considered satisfied only when its accumulated true time exceeds its duration time. The duration time prevents false alarms from transitory events or multiple alarms from events which are mostly continuous.

You can specify multiple conditions for an alarm. Each condition applies against the last ALARM command entered, and all such conditions must be satisfied at the same time before the alarm can begin.

Within Syntax Statements

THEN

This is an optional parameter, but if entered it specifies the action to be taken when the alarm is first satisfied. It will be repeated also if the alarm remains satisfied for a period longer than the specified REPEAT period.

If the first character of the action string is a colon (:), the action is interpreted as a single MPE command. Most MPE commands are allowed, including the :RUN command to run programs. The following commands are specifically not allowed:

```
ABORT    DO     HELLO     SETCATALOG
BYE      EOD    JOB       SHOWCATALOG
CHGROUP  EOJ    LISTREDO  REDO
DATA     EXIT   OPTION    RESUME
```

On MPE/iX, user-defined commands (UDCs) and user COMMAND-file commands may be entered. On MPE V, UDC and COMMAND-file commands are not allowed using this syntax.

If the first character of the action string is a dollar sign ($), the SCOPE data-collection program interprets the action as a command. Valid SCOPE commands are:

`$SCOPE NOTE` *message*	Logs a message to the GLOBAL log file. These messages can be retrieved using the UTILITY SCAN command.
`$SCOPE NEW`	Forces the SCOPE collector to process its PARM file again and act on any changes.

If the first character in the action is a caret (^), it is interpreted as the name of an ASCII file. This file will be opened, and the commands in it will be executed (subject to the restrictions listed above). This feature works on MPE V as well as on MPE/iX. See the discussion of Variable Substitutions in "Commands", later in this chapter.

Note Alarm actions will not be performed while logged data is being examined. They can be listed with substituted performance metrics if desired.

FINISH

This is an optional parameter, but if entered it specifies the action to be taken when an alarm is started and then ends (it is no longer satisfied). The options for this action are the same as those for the THEN parameter, above.

REPEAT=

[**MINUTES**] specifies the minimum time between repetitions of an alarm before it is allowed to be satisfied again. If the alarm is satisfied and remains satisfied for this time, it will REPEAT (it will process the THEN action again *without* performing the FINISH action). If the alarm is satisfied, finishes, and is satisfied again, no second THEN action will be taken until after the time specified by REPEAT=*nnn* following the initial satisfaction of the alarm. Alarm conditions continue to be processed during the REPEAT interval, but no THEN action is taken until the REPEAT interval passes and the alarm is again satisfied (no intermediate alarms are *remembered* during the repeat period).

Note The REPEAT time will always be greater than or equal to the longest DURATION time for any condition in the alarm. If you try to set a lower REPEAT time, it will be overridden. If you do not specify a REPEAT time, the default will be the time of the longest DURATION in the alarm.

VAR

The VAR command creates a user variable. You can have up to 26 user variables (named **A** through **Z**). Each variable definition consists of two items (*see* item1 and item2 definitions above) plus an arithmetic operation that is to be performed on them. The operations are:

+	User variable	=	item1 added to item 2
-	User variable	=	item1 minus item 2
*	User variable	=	item1 multiplied by item2
/	User variable	=	item1 divided by item 2

Within Syntax Statements

Note A user variable may include another user variable in its definition but variables are processed in alphabetical sequence from A through Z. For proper operation you should not reference a user variable *unless* it precedes the current one, alphabetically.

Table 7-1. Global Performance Items

Item Name	Description
TOTAL CPU	Total percentage of CPU busy.
SYSTEM CPU	CPU usage by system processes (percentage of total).
SESSION CPU	CPU usage by interactive session processes (percentage of total).
BATCH CPU	CPU usage by batch job processes (percentage of total)
MEM MGR CPU	CPU usage for memory management (percentage of total)
DISPATCH CPU	CPU usage for dispatching processes (percentage of total, MPE/iX only).
CACHING CPU	CPU usage for disc caching (percentage of total, MPE V only).
OTHER CPU	CPU usage for other activities such as ICS (percentage of total).
CPU PAUSED	Time CPU was idle *and* disc IO occurred (percentage of total).
IDLE CPU	Time CPU was idle and disc IO *did not occur* (percentage of total).
PHYS DISC	Physical disc IO rate (IOs/second), overall not counting MEM MGR.
SYSTEM DISC	Physical disc IO rate for system processes (IOs/second).
SESSION DISC	Physical disc IO rate for session processes (IOs/second).
BATCH DISC	Physical disc IO rate for batch processes (IOs/second).
MEM MGR DISC	Physical disc IO rate for memory management (IOs/second).
LOGICAL DISC	Total logical disc IO rate (IOs/second).
DISC UTIL	Average percentage disc utilization.
TRANSACT/MIN	Terminal transactions completed (transactions/minute).
FIRST RESP	Average first-response time for terminal transactions.
RESPONSE	Average response-to-prompt time for terminal transactions.
ANY LOGL IO	Highest logical disc IO rate on any disc drive (IOs/second; MPE V only).
ANY PHYS IO	Highest physical disc IO rate on any disc drive (IOs/second).
ANY MEM IO	Highest memory management IO rate on any drive (IOs/second).
ANY UTIL	Highest percentage utilization on any disc drive.
CPU QUEUE	Average number of processes waiting for or using the CPU.
DISC QUEUE	Average number of processes waiting for DISC transfers.
MEMORY QUEUE	Average number of processes waiting for MEMORY resources.
IMPEDE QUEUE	Average number of processes waiting for locks, etc..
NUM JOBS	Average number of batch jobs logged on.
NUM SESSIONS	Average number of sessions logged on.
ACTIVE JOBS	Average number of active batch jobs.
ACTIVE SESS	Average number of active sessions.

Within Syntax Statements

Performance metrics may be selected for a single application as defined in the PARM file. The syntax for specifying an application performance item is

```
application name:application item
```

where application name exactly matches one of the APPLICATION= strings defined earlier in the PARM file (including upper/lower case), and an application item is chosen from the list below.

Table 7-2. Application Performance Items

Item Name	Description
`TOTAL CPU`	CPU usage by this application's processes (percentage of total).
`PHYS DISC`	Physical disc IO rate for the application (IOs/second).
`TRANSACT/MIN`	Terminal transactions completed (transactions/minute).
`FIRST RESP`	Average first-response time for terminal transactions.
`RESPONSE`	Average response-to-prompt for terminal transactions.
`SERVICE LEVL`	Terminal transactions meeting service levels (percentage of total).
`CPU QUEUE`	Average number of processes waiting for or using the CPU.
`DISC QUEUE`	Average number of processes waiting for DISC transfers.
`MEMORY QUEUE`	Average number of processes waiting for MEMORY resources.
`IMPEDE QUEUE`	Average number of processes waiting for locks etc..
`CPU WAIT`	Time spent waiting for CPU resources (percentage of total).
`DISC WAIT`	Time spent waiting for disc data transfers (percentage of total).
`MEMORY WAIT`	Time spent waiting for main memory resources (percentage of total).
`LOCK WAIT`	Time spent waiting for software locks & impedes (percentage of total).
`IO WAIT`	Time spent waiting for IO other than disc and terminal input (percentage of total).

Variable Substitutions in Commands

You can substitute any performance item or user variable into a THEN or FINISH command before it is executed. To do so, enter an exclamation point (!) followed by the name of the item or letter of the user variable desired.

For example, if the global CPU value was *54.3*, the command

```
   :TELL MANAGER.SYS; The cpu is now !TOTAL CPU%
```

would execute as

```
   :TELL MANAGER.SYS; The cpu is now 54.3%
```

You can make multiple substitutions on a single command line. Commands found in a command file will also have variable substitutions performed on them before they are executed.

Caution Executing an MPE/iX COMMAND file using the `:filename` construction will *not* perform variable substitutions. To substitute variables, you must enter the file name preceded by a carat (^).

In addition to the performance metrics and user variables, you can use one of the following special items as a substitution value. Please note that these special items are *not* available for processing in user variables or alarm conditions, just in commands.

Substitution Code	Value Substituted
`!DATE`	The current date in MM/DD/YY format.
`!TIME`	The current time in HH:MM AM/PM format.
`!SYSTEM`	The system ID string from the SCOPE collector.

Within Syntax Statements

Examples

```
    VAR A = SYSTEM DISC + SESSION DISC
    VAR B = A + BATCH DISC              (All user disc IO.)

  ALARM="Session CPU is too High" TYPE=CPU SEVERITY=5
    IF SESSION CPU > 70 FOR 20 MINUTES
    THEN :TELLOP Session CPU = !SESSION CPU%
    FINISH :TELLOP Session CPU now down to !SESSION CPU

  ALARM="Disc Rate Bug" TYPE=DISC SEVERITY=10

  IF PHYS DISC < VAR B FOR 60;          (Compare physical disc vs user disc.)
    THEN :STREAM DCPJOB.COLLECT.HPPROBE
    REPEAT=1440                         (But don't do it more than once a day.)

  ALARM="Once an Hour"
    IF 2 > 1 FOR 60                     (This condition is ALWAYS true.)
    THEN :TELLOP !DATE !TIME !SYSTEM Response = !RESPONSE

    REPEAT=60                           (Do this once an hour.)

  ALARM="Bad Programmers" TYPE=punish SEVERITY=99
    IF HPDESK :RESPONSE > 5 FOR 10 MINUTES
    IF PROGRAM DEVELPMENT :CPU > 40 FOR 10 MINUTES
    THEN :ALTACCT DEVELOP :CPU >  0 :CONNECT=0
```

Using Performance Collection Software Alarms

To use an alarm, do the following:

1. Create a PARM file with the desired alarms and user variables.
2. Run the UTILITY program.

   Enter the PARMFILE command and specify the name of the PARM file containing the alarm definitions. If any errors are reported, correct them *before* proceeding. You may specify the Performance Collection Software log file start and end dates, etc., then enter the SCAN command. During the SCAN function, each data sample is processed against the alarms defined in the PARM file. If DETAIL=ON then the beginning, ending, and actions of each alarm will be printed as they are triggered. Variable substitution will be performed on any THEN or FINISH actions and the result printed but *the actions will not be performed during the UTILITY SCAN function.*

   At the end of the SCAN function, a summary will be printed showing the number of times each alarm was triggered and the total number of minutes each was active. If you set ALARMS=OFF, no alarm events will be listed but a summary still will be produced.

Using UTILITY to Filter Log Files

You can use the UTILITY program to process alarms against an existing Performance Collection Software log file to decide if the log file contains situations that you will want to study in more detail. While it is easy to perform this operation interactively, as described earlier, it might be best to perform this task in a more automated fashion.

To facilitate the batch capability of Performance Collection Software alarms, the UTILITY program will save the alarm summary results in Job Control Words (JCWs) which can be interrogated by standard MPE Job Control Language.

Two Job Control Words are created and set for each alarm defined in the PARM file.

- RXALARMCOUNT*n* is set to the number of times the alarm was triggered or was repeated (*n* is the sequence number of the alarm in the PARM file).
- RXALARMTIME*n* is set to the number of minutes for which the *n*th alarm was active.
- RXSCANHOURS is set to indicate the total number of hours Performance Collection Software data was scanned.

Note The individual alarms need not be listed during the SCAN in order to set the JCW values.

The following batch job will (1) extract the last week's data from the raw log files, (2) scan it using the UTILITY program, then (3) DSCOPY it to a central system for analysis *if* it is interesting.

```
!JOB SCOPE.SYS,SCOPE
!RUN EXTRACT                 (Extract the last week's data)
LOG LOGGLOB.SCOPE.SYS
START TODAY-7
OUTPUT RXLOG,PURGE
GLOBAL BOTH
APPLICATION BOTH
PROCESS DETAIL
DISCSPACE DETAIL
EXTRACT
EXIT
!RUN UTILITY                 (Now scan the extracted file.)
LOG RXLOG
PARM PARMALRM                (PARMALRM contains alarm definitions.)
DETAIL OFF                   (Save paper, don't list details.)
ALARMS OFF                   (Don't list individual alarms.)
SCAN                         (Scan the log file.)
EXIT                         (UTILITY is done, the JCWs have been set.)
!SETJCW INTERESTING=0        (Set my own JCW to "uninteresting.")
!IF RXALARMCOUNT1>0 THEN     (If alarm 1 went off even 1 time.)
! SETJCW INTERESTING=1       (I am interested.)
!ENDIF
!IF RXALARMTIME2>60 THEN     (IF alarm 2 was ON for more than one hour.)
! SETJCW INTERESTING=2       (I am interested.)
!ENDIF
!IF INTERESTING>0 THEN       (IF I am interested.)
! REMOTE HELLO SCOPE.SYS,SCOPE;DSLINE=CENTRAL (Ship it to central.)
! DSCOPY RXLOG TO THISONE,CENTRAL;MOVE
! REMOTE BYE
!ENDIF                       (Otherwise do nothing.)
!EOJ
```

8

What's New: Changes to Performance Collection Software

The first section of this chapter is intended for current users of HP LaserRX/MPE *version A*. It contains an overview of the enhancements made to *version A* to create Performance Collection Software *version B.00.00*. Details of these changes were covered in preceding chapters.

The changes and enhancements made to Performance Collection Software since the B.00.00 release are described in the second section of this chapter.

Changes from Version A to Version B.00.00

Command-Driven User Interface/Changes to Batch Files

The command-driven user interface for UTILITY and EXTRACT replaces the prompt-and-answer dialog used in earlier versions of these programs.

If you are using these programs in *batch* mode, you *must* alter your batch jobs to include the appropriate command syntax. See chapters 4 and 5 for more information.

Note Significant enhancements to the UTILITY and EXTRACT programs make it easier to conduct unattended batch operations. (For examples, see the WEEKLY, MONTHLY, and YEARLY EXTRACT commands.)

New Log File Formats

Log file formats have changed to add more performance metrics. These changes require that existing log files be converted to the new format before being used by this release of Performance Collection Software. See chapter 4 for details on the CONVERT command.

Caution Your existing log files will be altered by the conversion process. If you did not back up your log files when you installed this release, you should back them up *before* you convert them. See chapter 1.

Disc Space Metrics

With this release of Performance Collection Software, you can elect to log disc space information by adding the DISCSPACE parameter to the LOG command in the PARM file. A new log file named LOGDISC is created when initial DISCSPACE logging takes place..

Disc space information is captured and logged once a day. The new DAILYTIME command lets you choose the time of day.

A new program in the SCOPE.SYS group allows you to log disc space information. The name of the disc collector you use depends on your host operating system:

- For MPE V, use SCOPE2.
- For MPE/iX, use SCOPEXL2.

Changes to the PARM File

You need not change your PARM file unless you want to log disc space information or take advantage of the new THRESHOLD parameters or the new WAIT THRESHOLD directive. The THRESHOLD parameter's default values are listed in table 3-2.

WAIT THRESHOLD

Use this directive to mark a process as *interesting* and log it if it spends too much of time *waiting* for a key system resource.

The THRESHOLD directive can log a process that is *using too much* of a resource, whereas the WAIT THRESHOLD directive can log processes that are *waiting too long* to get access to a resource. This allows you to log the cause and effect of a system bottleneck.

Parameters of the WAIT THRESHOLD directive follow.

CPU, DISC, MEMORY, IMPEDE. Use these parameters to specify the percentage of the 1-minute sample interval that a process must wait for a resource (CPU, Disc, Memory, or Impede) before being logged.

New THRESHOLD Parameters

New THRESHOLD parameters follow.

NONEW, NOKILLED. Use these parameters to prevent the logging of processes that are interesting *only* because they are new or killed but are otherwise uninteresting.

NOSHORT. Use the NOSHORT parameter to reduce the disc space used for SHORT processes. A SHORT process is one that is created and terminated within a specified interval. Since a system might run many short processes each day, these processes could occupy a significantly large part of the process log file.

MINTHINK, MAXTHINK. Use these parameters to fine-tune the algorithm SCOPE(XL) uses to filter terminal transactions.

MINTHINK specifies the minimum user think time necessary to eliminate hardware-generated transactions, such as terminal status reads.

MAXTHINK filters out the first transaction following a prolonged absence from the terminal, since such transactions might not be representative.

Generally, you should not alter MINTHINK or MAXTHINK without careful planning, since you might drastically alter the transaction rates and response times reported by Performance Collection Software.

QUEUE. Use this parameter to select processes for an application based on the dispatcher queue (L, A, B, C, D, E) in which they are executing.

This is used in addition to the selection by program name, job or session type, and user logon.

Within Syntax Statements

OR. Use this parameter to apply more than one application definition to the same application. This gives you more flexibility in defining the processes that belong to a given application.

DISCGROUP, GROUP. Use these parameters to define disc-space logging groups. If no disc groups are defined, disc space will be logged at the account level.

SCOPE(XL) Changes

The functions of SCOPE(XL) remain the same, but the log files SCOPE(XL) creates are not backward compatible with HP LaserRX/MPE, version A.

Many new performance and system-management metrics are logged without a significant increase in disc space used or the CPU overhead. The SCOPE.SYS group can contain the following new files:

`LOGDISC`	This disc space log file is created whenever the LOG DISCSPACE command is found in the PARM file. It can also be created using UTILITY, if you choose a size other than the default.
`STATUS`	This file appears automatically to log noteworthy events in the life of SCOPE(XL): starting, stopping normally or abnormally, resizing a raw log file, etc.. The file is a circular ASCII file and can be printed using the FCOPY command or the MPE/iX PRINT command. The file can be resized to hold more or less data, but be careful to preserve its characteristics (circular, record length, and other parameters).
`SCOPE2`	This is the daily disc space collection program for MPE V.
`SCOPEXL2`	This is the daily disc space collection program for MPE/iX.
`SCOPEIN`	This is a message file used for communicating between the UTILITY and SCOPE(XL) programs.
`SCOPEOUT`	This message file appears whenever UTILITY is awaiting a response from SCOPE(XL). It is purged automatically when the UTILITY program terminates. The SCOPEOUT file is created in the LOGON group of the user that runs UTILITY. It can be purged manually, if it remains after UTILITY is executed.

`HCLS`*nnnn* `HCLST`*nnn*	These files are created when a remote connection is made by Performance Collection Software or when the UTILITY program's VERSION command is executed. These files will be created in the LOGON group and can be purged at any time. A future release of cooperative services should eliminate the creation of these files.

UTILITY Changes

The following enhancements are made to the UTILITY program's functions:

- You no longer have to *scan* log files before *resizing*.
- You can specify log file resizing in days, in megabytes, or both.
- More information is reported while scanning a log file:
  - You can list application definitions and names in a detailed report.
  - You can generate an application summary report showing the number of application records plus the percentage of the total CPU, disc, and terminal transactions for each application defined.
  - You can generate a process summary report showing the number of process records logged for each interest reason. You can use this report to fine-tune the PARM file THRESHOLD and WAIT THRESHOLD directives.
  - You can scan a portion of a log file by specifying the start and stop dates for SCAN.
  - You can display additional log file parameters and track their changes during a detailed SCAN. These new parameters include:

    Type of host system (3000/950, 3000/68, and so on).
    System serial number (or software serial number on MPE/iX).
    Operating system release (A.30.00).
    Total size of main memory.
    Number of processors (for multiprocessor MPE/iX systems).
    New parameters for the THRESHOLD and WAIT THRESHOLD directives in the PARM file.
  - You can list user- and SCOPE-generated notes.

Within Syntax Statements

- You can use the CREATE command to create individual log files and specify their size in days or megabytes.

New UTILITY functions include:

- Command-driven user interface.
- Online help.
- Terminal softkey support.

The new functions enable you to:

- Direct reports to another output device.
- Use the CONVERT command to convert raw and extracted log files from the format of the earlier release to that of the current release.
- Examine a PARM file and report any errors.

  This report also indicates how much room is left for defining applications. The resulting parameters can be listed, including any default values not specified in the PARM file.
- Execute MPE commands without leaving the UTILITY program.
- Use the VERSIONS command to print the version numbers of all the host Performance Collection Software files. It can also report any critical programs that are missing.
- Communicate with SCOPE(XL) using the UTILITY program. You can perform the following actions:
  - Determine whether SCOPE(XL) is running. If it is running, then print out the current PARM file parameters and the percent-full values for the global, application, and process files.
  - Stop SCOPE(XL), and verify that it has terminated.
  - Use the SCOPE NOTE command to send a user note to the global log file. This note can be printed during the SCAN operation. It stays with the data even when it is extracted.
  - Have SCOPE(XL) resample its PARM file without stoping and restarting the collection process.

    Any PARM file value can be changed *except* the system ID.

EXTRACT Changes

EXTRACT and UTILITY have the same user interface. This new interface includes online help, softkey support, and the ability to redirect extract reports.

The basic EXTRACT function remains unchanged: to extract data from raw or extracted log files, optionally subset or summarize the data extracted, and write it to a new extracted log file.

Existing functions are enhanced as follows:

- Date and time formats can be customized to other languages using MPE's native language support (NLS) features.
- Disc space data can be extracted.
- MPE commands can be executed without leaving EXTRACT.
- The process used to append data to an existing extracted log file was reworked significantly, maintaining the integrity and viability of the resulting log file. Maximum user flexibility is allowed as long as it does not produce a log file that is misleading or that can cause display errors.

New EXTRACT functions include the following:

- You can extract data using the WEEKLY, MONTHLY, and YEARLY commands.

  These functions give the extracted log file a unique name for each week, month, or year, and match the start and end of the extraction to the calendar week, month, or year. These functions simplify unattended remote extractions and help in archiving Performance Collection Software log file data.

Changes from Version B.00.00 to Version B.00.02

The changes and enhancements made to Performance Collection Software *since* the B.00.00 release are described here.

Starting with MPE/iX B.40.00 (4.0) and MPE V G.23.00 (23) the Performance Collection Software can be installed or updated as a normal part of the MPE update procedure. Using the MPE update procedures should make it easier to install. Time-consuming uploads from the PC will not be needed, and you will be less likely to get a mismatch between the collector and MPE versions.

Changes in SCOPE(XL)

On MPE V, the limit of 628 processes was raised to 1024 on the release 23 MI. SCOPE B.00.02 will allow you to collect performance data on systems with more than 628 processes.

SCOPEXL B.06.02 now supports MPE/iX release 4.0. This new SCOPEXL collector was adjusted for all internal changes in the operating system, and these changes should be transparent to Performance Collection Software users.

A new Service Level Agreement metric has been added to both MPE V and MPE/iX versions of SCOPE(XL). This metric measures the percentage of terminal transactions that were below specified response times. Different service levels may be set for each application. See appendix B for more information.

Log file maintenance and roll back procedures can be scheduled to occur at specified times instead of only when a log file is filled. SCOPE(XL) will examine each log file once a day. If the program estimates that the file might become filled within the next 24 hours, it will initiate a roll back immediately. Refer to the MAINTIME directive for more information.

A new report can be generated to help identify the users responsible for disc space usage and to test the DISCGROUP definitions in the Performance Collection Software PARM file. You may generate this report at any time by running the SCOPE2 or SCOPEXL2 program and specifying `;INFO="HELP"`.

Default thresholds have been adjusted:

Threshold	**Old Value**	**New Value**
`MINTHINK`	0.1	0.2
`WAITCPU`	100	50
`WAITDISC`	100	50
`WAITIMPEDE`	100	25
`WAITMEMORY`	100	25

Host Access to Data

A new command, EXPORT, was added to the existing EXTRACT program. This command will create files containing selected Performance Collection Software data in formats suitable either for printing or for further analysis by other programs. By permitting direct access to Performance Collection Software data on the host HP 3000 system, it will be easier to do automated and custom reporting of performance data.

Performance Alarms

Historical analysis of existing Performance Collection Software log files can be performed using the UTILITY program.

Special features added to the UTILITY program allow you to use the alarm-processing feature to examine the log files in a routine batch job. Action is taken only when system performance indicates it is necessary.

Changes in EXTRACT and UTILITY

You can specify starting and stopping dates relative to the current date by using a special `TODAY` or `TODAY-`*nnn* syntax in the START and STOP commands.

A

MPE/iX Metric Information

Introduction

The MPE V and MPE/iX operating systems and measurement interfaces differ and as a result, performance data collected on the two systems differ.

These differences are discussed below:

- MPE/iX Measurement Differences.
- MPE/iX versus MPE V Data Collection.

MPE/iX Measurement Differences

Some performance metrics are not available on all releases of the MPE/iX operating system. SCOPEXL is the Performance Collection Software data collector for MPE/iX systems. The way SCOPEXL calculates values for these metrics is described in the following sections.

Individual Disc I/Os by Device

Counts of individual disc I/Os by device are not available at the application level, therefore the total disc I/Os for the Application Detail data (zoom-by-application) are accumulated into a single disc device.

On MPE/iX systems, counts of logical disc I/Os for an individual device will be available in a future release of the operating system. Until then, these values are set to zero on GLOBAL DISC DETAIL graphs.

MPE/iX versus MPE V Data Collection

The collector programs for MPE/iX (SCOPEXL) and MPE V (SCOPE) have identical functions. The programs have different names to prevent a user from executing the wrong version of the collector on a system.

You can use the same PARM files to define applications, identify systems, and set thresholds. The log files created on both systems have the same names: LOGGLOB, LOGAPPL, LOGPROC, LOGDISC, and LOGALRM.

An additional log file, LOGINDX, is created with MPE/iX. It is very small and can appear to be empty (i.e. contain zero records). *Do not purge it!* The file contains the information necessary for rapid positioning into the other three log files.

All log files are created by SCOPEXL when it is first run or are created by the UTILITY program. Since the data collected by SCOPE and SCOPEXL are compatible, you can expect the log files to grow at about the same rate (usually at an average rate of about 1 megabyte of data per day).

MPE/iX allows 31 user-defined applications while MPE V allows only 15. There is a limitation when you move a raw or extracted MPE/iX log file to an MPE V system. EXTRACT will be able to extract application *details* for all applications, but will only extract application *summaries* for the first 16 applications. You will receive a warning message if you try to use the MPE V EXTRACT program to extract application summaries of a log file with more than 16 applications.

MPE/iX provides 5 kilobytes of disc space for application definitions; MPE V provides 2 kilobytes of disc space.

Logical Disc I/Os

On MPE/iX systems, there is a relatively higher ratio of logical disc I/O to physical disc I/O than on MPE V systems. This is partly due to MPE/iX caching disc writes *and* reads, but to a larger degree, it is due to the two systems defining logical I/O differently.

Under MPE V, a logical I/O occurs when the file system requests the *I/O system* to transfer a block of data to or from a disc file. Under MPE/iX, a read attempt or write attempt occurs whenever a program requests the *file system* to transfer a block of data.

These transfers can occur at each record, at each block (if multirecord I/O is done), or not at all (if mapped files are used). The metric that records MPE/iX read/write attempts is comparable to the metric that records MPE V logical I/Os, but they are *not* equivalent.

B

Service Level Agreements

This enhancement to the SCOPEXL performance data collector adds a new application metric to an existing field of the application record. No log file format changes are required.

This metric can be used for service level agreements that contract for a certain percentage of terminal response times to be *at* or *below* a given level. For example, an agreement might specify that 95% of the terminal response times for the Order Entry application will be 1 second or less.

Some applications base service levels on the first-response time, others use the response-to-prompt times. This enhancement allows you to use either method or a combination of methods.

There are several rules:

1. The system manager must enter the agreed-upon response time values into the PARM file for those applications of interest. Each application may have different service levels, if desired.
2. The collector will examine all of a process's applications once a minute. If the process's average response time exceeds the first-response or response-to-prompt service level value for that interval, all transactions performed by that process are considered as "failing to meet the service level agreement."
3. At the end of each five-minute collection period, the number of transactions that met the service level agreement—the application's first-response *and* response-to-prompt times were at or below the agreed upon values—is logged as a percentage of the total transactions in the application.

Within Syntax Statements

You can easily assess conformance to a service level agreement by plotting the percentage of transactions that met the agreed upon values over time. A service-level violation occurs each time this percentage falls below the threshold value.

By monitoring the percentage value *before* it drops below the threshold value, you might be able to avoid violations of the agreement.

Using the Facility

You must remember that *all* transactions for a process are considered to fail the service level agreement if *either* the first-response *or* the response-to-prompt time exceeds the specified values. If you want to use only *one* criterion—such as response-to-prompt time—you should set the *other* criterion (in this case, first-response) to a *large value*. For example, to use a response-to-prompt service level of 1 second, set RESPONSE=1 and FIRST=3000.

Set the values by adding the following line *after* the desired APPLICATION line in the PARM file:

```
SERVICE [LEVEL] FIRST=seconds RESPONSE=seconds
```

If service levels are not set for an application, it will use the interesting process threshold values for `FIRST` and `RESPONSE`. If only one service level threshold is set, the other will not be used.

By default, processes that violate a service level agreement are considered *interesting* and will be logged even if they are not interesting for any other reason. The Interest Reason for these processes will be set to `FIRST` or `PROMPT` depending on which thresholds were violated.

Recommendations

1. If only one application is included in the agreement, specify the `SERVICE LEVEL FIRST=` and `RESPONSE=` for that application.
2. If all or a majority of the applications are included in the agreement, specify high interesting process thresholds values—such as `THRESHOLD RESPONSE=3000 FIRST=3000`—then set the service level thresholds for each application. In this way, processes will be considered interesting for `FIRST` or `PROMPT` *only if* they violate the corresponding service level.
3. If you want all applications on the system to have the same service level,—for instance, if the agreement specified all users of the system rather than specific applications—set the interesting process thresholds to the desired values. You need not set individual application levels.

C

UTILITY SCAN Report Details

There are three phases to a UTILITY SCAN report:

- Initial values.
- Chronological details.
- Summaries.

DETAIL, NOTES, and ALARMS Commands

Table C-1 shows what information is printed in all SCAN reports and in reports printed with DETAIL=ON, NOTES=ON, or ALARMS=ON commands.

Table C-1. Information Contained in SCAN Reports

Initial Values:	
Initial PARM file global information.	Printed *only* if DETAIL=ON
Initial PARM file application definitions.	Printed *only* if DETAIL=ON
Chronological Detail:	
PARM file global change notifications.	Printed *only* if DETAIL=ON
PARM file application addition/deletion notifications.	Printed *only* if DETAIL=ON
Collector OFF time notifications.	Printed *only* if DETAIL=ON
Application-specific summary reports.	Printed *only* if DETAIL=ON
Collector-generated notes (log file resize, Collector shutdown, and so on).	Printed *only* if NOTES=ON
USER-generated notes (entered through UTILITY's SCOPE NOTE command).	Printed *only* if NOTES=ON
Alarm BEGIN/END/REPEAT events.	Printed *only* if ALARMS=ON
Alarm actions.	Printed *only* if ALARMS=ON *and* DETAIL=ON
Summaries:	
Process log reason summary.	Printed *only* if process data was scanned.
SCAN START and STOP actual dates and times.	Always printed.
Application overall summary.	Printed *only* if application data was scanned.
Performance alarm summary.	Printed *only* if PARMFILE contained alarm definitions.
Collector coverage summary.	Always printed.
Log file contents summary.	Includes space and dates covered.
Log file empty space summary.	Always printed.

Initial Values

This section contains examples of the following initial values report phases:

- Initial PARM file global information.
- Initial PARM file application definitions.

Initial PARM File Global Information

You must specify DETAIL=ON.

This report lists the contents of SCOPE's PARM file at the time of the earliest global record in the log file. Later global-information change notifications are based on the values in this report. If no change notification exists for a particular parameter, it means that the parameter kept its original setting for the duration of the scan.

```
01/03/90 12:36 SYSTEM ID="COOKIE PTC Hewlett-Packard SERIES 950"
SCOPE/XL B.00.00.00 SAMPLE INTERVAL = 300,300,60 SECONDS, LOG VERSION=B
OS=MPE/XL     A.30.00    SYSTEM=3000/950   S/N=0
LOGGING GLOBAL APPLICATION PROCESS DISCSPACE RECORDS
THRESHOLDS: CPU= 10%, DISC=10/SEC, RESP=5.0 SEC, FIRST=1.0, SEC TRANS=100
            MINTHINK=0.001 SEC, MAXTHINK=3600 SEC
            NONEW=FALSE, NOKILLED=FALSE, NOSHORT=30
WAIT THRESHOLDS: CPU=100%, DISC=100%, MEMORY=100%, IMPEDE=100%
MEMORY: PHYSICAL=128.0 MBYTES,  USER=128.0 MBYTES, SWAP=0.0 MBYTES
DISC LOGICAL DEVS:     1     2     3     4     5    15    16    17
```

The date and time listed on the first line correspond to the *first date and time* in the global log file and indicate when SCOPE was started. Data records may have been rolled out of the global log file so the date and time on this report do not necessarily indicate the *first global record* in the log file.

Initial PARM File Application Definitions

You must specify DETAIL=ON and have APPLICATION data in the log file.

This report lists the name and definition of each application at the time the first application record is listed in the log files. Any application addition or deletion notifications you receive are based on this initial list of applications.

```
01/03/90 12:36 APPLICATION(1)="OTHER"

01/03/90 12:36 APPLICATION(2)="Resource Sharing"
 FILE=@.PPC.SYS; @.PSPOOLER.SYS; PCLINK.PUB.SYS; PCSERVER.PUB.SYS;
 FILE=PSUTIL.PUB.SYS; PDSERVER.PUB.SYS

01/03/90 12:36 APPLICATION(3)="SPOOLING"
 USER=@,RSPOOL@.SYS,@; RSPOOL@,@.@,@
```

Note

During the SCAN, you can be notified of applications that were *added* or *deleted*. This decision is based entirely on the *application name*.

No attempt is made to detect a change in the *definition* of an application. If an application with a new name is detected, it is listed along with its new definition.

The date and time on this record is the last time SCOPE was started before logging the first application record currently in the log file.

Chronological Detail

This section contains examples of the following chronological detail report phases:

- PARM file global change notifications.
- PARM file application addition and deletion notifications.
- SCOPE OFF time notifications.
- Application-specific summary report.
- SCOPE-generated notes.
- USER-generated notes.
- ALARM events.

PARM File Global Change Notifications

You must specify DETAIL=ON and have GLOBAL data in the log file.

This report can be generated any time SCOPE is started or is instructed to resample its PARM file. If the current PARM file collection parameters differ from the parameters when SCOPE ran last, a global change notification can occur.

The following is an example of the change notifications that occur when four new disc drives are added to the system.

```
01/26/90 16:43 THE NUMBER OF DISC DRIVES CHANGED FROM 8 TO 12
01/26/90 16:43  DISC #  9 WAS ADDED AS LDEV 18
01/26/90 16:43  DISC # 10 WAS ADDED AS LDEV 30
01/26/90 16:43  DISC # 11 WAS ADDED AS LDEV 31
01/26/90 16:43  DISC # 12 WAS ADDED AS LDEV 32
```

PARM File Application Addition/Deletion Notifications

You must specify DETAIL=ON and have APPLICATION data in the log file.

User-defined applications can be added or deleted each time SCOPE is started or is instructed to resample its PARM file. If you find an application name that does not match the last set of applications, you can list an application addition, deletion, or change notification. If the name of an application has not changed from that previously reported, it is not listed again.

Note Application definitions are not checked for changes at this time. They are listed when an application name is changed, but any change to an existing application's definition without an accompanying name change is not detected.

```
01/31/90 21:11  APPLICATION  4 "COMPILES           " WAS ADDED
  FILE=COBOL@.PUB.SYS; FORTRAN.PUB.SYS; FTN@.PUB.SYS; SPL
  FILE=PASCAL.PUB.SYS
```

This example indicates a new application was started.

SCOPE OFF Time Notifications

You must specify DETAIL=ON.

If an extracted files contains only summary information, times are rounded to the nearest hour.

```
01/29/90 11:00 - 01/29/90 12:34 COLLECTOR OFF (    01:34:04)
```

The first date and time (`01/29/91 11:00`) indicate the last valid data record in the log file before SCOPE was restarted. The second date and time (`01/29/91 12:34`) indicate when SCOPE was restarted.

The last field (in parentheses) shows how long SCOPE was *not* running. The format is *ddd/hh:mm:ss*, where *ddd* are days, and *hh:mm:ss* are hours, minutes, and seconds. Zeros to the left are deleted.

In this example, SCOPE was off on January 29, 1991 between 11:00 AM and 12:34 PM. The summary information shows that data was not collected for 1 hour, 34 minutes, and 4 seconds.

Application-Specific Summary Report

You must specify DETAIL=ON and have APPLICATION data in the log file.

This report can help you define applications. Use the report to identify applications that are accumulating either too many or too few system resources and those that could be consolidated with other applications. Applications that accumulate too many system resources might benefit by being split into smaller pieces.

You should define applications in a way that helps you make decisions about system performance tuning. It is unlikely that system resources would accumulate evenly across applications.

The application-specific summary report is generated whenever the application definitions change to allow you to access the functionality of the applications before and after the change.

A final report is generated for all applications. This report covers only the time since the last report and not the entire time covered by the log file.

```
                                      PERCENT OF TOTAL
APPLICATION            RECORDS   CPU   DISC  TRANS
-------------------- ---------- ------ ------ ------
OTHER                     22385  45.7%  20.9%  63.0%
Resource Sharing           7531   6.0%   2.2%  17.1%
SPOOLING                  13813   2.4%   0.3%   0.0%
ON-LINE COMPILES          13119   2.9%   1.7%   0.1%
BATCH COMPILES             8429   2.9%   0.1%   2.2%
ORDER ENTRY                 387   0.1%   0.0%   0.0%
ELECTRONIC MAIL            6251   3.8%   1.3%   9.6%
PROGRAM DEVELOPMENT        3141   9.1%   2.4%   0.6%
RESEARCH DEPARTMENT        3968   8.7%   2.0%   6.0%
BILL OF MATERIALS           336   0.6%   0.2%   0.1%
FINANCIALS                 1080   5.0%   1.5%   0.5%
MARKETING DEPT             2712  12.9%  67.3%   0.0%
GAMES                       103   0.1%   0.0%   0.6%
-------------------- ---------- ------ ------ ------
ALL USER APPLICATIONS     73.1%  54.3%  79.1%  37.0%
```

Within Syntax Statements

SCOPE-Generated Notes (Collector Shutdown)

You must specify NOTES=ON (or NOTES=102) and have GLOBAL data in the log file.

```
01/30/90 10:56 NOTE [102] SHUTDOWN REQUESTED BY PURGING "RUN" FILE
```

SCOPE can enter a note record in the global log file. Typically, it does this to indicate an important event such as its voluntary termination. In the example above, SCOPE notes that it is terminating voluntarily because the RUN file was purged.

The number in brackets (`[ ]`) indicates the origin of the note and can be filtered from this listing using the LEVEL option of the NOTES command.

User-Generated Notes

A note can be entered in the global log file by running the UTILITY program and issuing the SCOPE NOTE command. These notes are displayed during a SCAN of the log file if the NOTES=ON or NOTES=100 command is in effect.

```
04/01/90 08:05 NOTE [100] SYSTEM MELT DOWN IN PROGRESS
```

Performance Alarm Events

Before you can see any performance alarm events, you must define alarms in a file that can be accessed using the PARMFILE command. See chapter 7 for more information on defining alarms. The log files do not contain these events, but the events are generated based on log file data following the rules in the PARM file. If you use a different set of alarm definitions during a SCAN, you will obtain a different set of alarm events.

If alarms are defined *and* you have `ALARMS=ON`, an *alarm-begin* event will be listed every time an alarm has met all its conditions for the specified duration. When these conditions are no longer satisfied, an *alarm-end* event will be listed. If an alarm condition is satisfied for a period long enough to generate another alarm without having first ended, a *repeat* event will be listed.

Each event listed will show the alarm number, how long the alarm has been active, and the alarm ID. For example:

```
11/21/91 21:30 ALARM[ 4] BEGIN  AFTER 10 MIN "MEMORY THRASHING"
11/21/91 21:40 ALARM[ 4] END    AFTER 20 MIN "MEMORY THRASHING"
12/05/91 21:20 ALARM[ 5] BEGIN  AFTER 10 MIN "CPU OVERLOAD"
12/05/91 21:30 ALARM[ 5] REPEAT AFTER 20 MIN "CPU OVERLOAD"
12/05/91 21:35 ALARM[ 5] END    AFTER 25 MIN "CPU OVERLOAD"
```

If you specified `ALARMS=ON` and `DETAIL=ON`, you will see the actions (`THEN` and `FINISH`) in addition to the alarm events. The actions *will not be performed*, but they will be listed with any requested parameter substitutions in place.

```
11/21/91 21:30 ALARM[ 4] BEGIN  AFTER 10 MIN "MEMORY THRASHING"
:TELLOP Severe Memory Shortage (Swapping at 25.8/second)
11/21/91 21:40 ALARM[ 4] END    AFTER 20 MIN "MEMORY THRASHING"
:TELLOP OK, relax, swapping is down to 2.3/second now.
```

Summaries

This section contains examples of the following summary report phases:

- Process log reason summary.
- SCAN START and STOP actual dates and times.
- Application overall summary.
- SCOPE coverage summary.
- Log file contents summary.
- Log file empty space summary.

Process Log Reason Summary

You must have PROCESS data in the log file.

This report helps you set the interesting process thresholds for SCOPE. The report lists every reason a process might be considered interesting, and thus get logged, along with the total number of processes logged that satisfied each condition.

The following is an example of a process log reason summary report:

```
PROCESS SUMMARY REPORT 03/13/90 10:08 AM TO 03/30/90 8:48 AM
THERE WERE 382.6 HOURS OF PROCESS DATA
PROCESS RECORDS WERE LOGGED FOR THE FOLLOWING REASONS:

LOG REASON      RECORDS   PERCENT   RECS/HR
--------------- -------   -------   -------
NEW PROCESSES     20276     14.6%      53.0
KILLED PROCESSES  19281     13.9%      50.4
CPU THRESHOLD     15338     11.1%      40.1
DISC THRESHOLD     1765      1.3%       4.6
TRANSACTIONS        226      0.2%       0.6
FIRST RESPONSE    10190      7.3%      26.6
RESPONSE TIME      6025      4.3%      15.7

WAIT ON CPU        3817      2.8%      10.0
WAIT ON DISC       1459      1.1%       3.8
WAIT ON MEMORY     1104      0.8%       2.9
WAIT ON IMPEDE     7455      5.4%      19.5

SHORT PROCESSES      27      0.0%       0.1

NOTE:  A PROCESS MAY BE LOGGED FOR MORE THAN ONE REASON AT A TIME.
       RECORD COUNTS AND PERCENTAGES DO NOT ADD UP TO 100 PERCENT
       OF THE PROCESS RECORDS.
```

If DETAIL=ON, this report is generated each time a threshold value is changed so you can evaluate the effects of that change. Each report covers the period since the last report. A final report, generated when the scan is finished, covers the time since the last report.

If DETAIL=OFF, then only one report is generated covering the entire scanned period.

You can reduce the amount of process data logged by SCOPE by raising the thresholds of the interest reasons that generate the most process log records. To increase the amount of data logged, lower the threshold for the area of interest.

In the previous example, you can decrease the amount of disc space used for the process data (at the expense of having less information) by raising the CPU threshold or setting the NONEW threshold.

If you want more information on processes that are getting poor response time, you can lower the threshold for RESPONSE or FIRST.

Within Syntax Statements

The last category, SHORT PROCESSES, is slightly different from the others. It lists the number of processes that were logged but that might not have been logged if the NOSHORT threshold had been used.

SCAN Start and Stop

This summary report will be printed if any valid data was scanned. It gives actual dates and times that the SCAN was started and stopped.

```
SCAN STARTED ON        01/03/90 12:40 PM
SCAN STOPPED ON        05/11/90  1:25 PM
```

Application Overall Summary

You must have APPLICATION data in the log file.

This report is an overall indicator of how much system activity is accumulated in user-defined applications, rather than in OTHER applications. If a significant amount of a critical resource is not being captured by user applications, you might consider scanning the process data for processes that can be included in user applications.

```
OVERALL, USER DEFINED APPLICATIONS ACCOUNT FOR
    82534 OUT OF    112355 RECORDS     ( 73.5%)
    218.2 OUT OF     619.4 CPU HOURS   ( 35.2%)
     24.4 OUT OF      31.8 M DISC IOS  ( 76.8%)
      0.2 OUT OF       0.6 M TRANS     ( 27.3%)
```

Performance Alarm Summary

You must use the PARMFILE command to specify process alarm definitions. Setting ALARMS and DETAILS does not affect the generation of the Performance Alarm Summary. See chapter 7, “Performance Alarms,” for more information.

Table C-2. Performance Alarm Summary

Alarm	Count	Minutes	Description
1	1	25	CPU is monopolized by sessions.
2	1	15	Batch is starved for CPU.
3	0	0	Disc utilization.
4	9	80	Memory thrashing.
5	0	0	Runaway terminal.

Collector Coverage Summary

This report is printed if any valid global or application data was scanned. It indicates how well SCOPE is being used to capture system activity. If the percentage of time SCOPE was off is high, as in the example below, then you should review your operational procedures for starting and stopping SCOPE.

```
THE TOTAL TIME COVERED WAS       108/16:14:51 OUT OF 128/00:45:02
TIME LOST WHEN COLLECTOR WAS OFF  19/08:30:11  15.12%
THE SCOPE COLLECTOR WAS STARTED           45 TIMES
```

This report will be more compete if global detail data is included in the scan. If only summary data is available, you can determine the time SCOPE was stopped and started only to the nearest hour. (An appropriate warning message is printed with the report if this is the case.)

The total time covered is determined by accumulating all the interval times from the logged data. The `OUT OF` time metric is calculated by subtracting the starting date and time from the ending date and time. This should represent the total time that could have been logged. The `TIME LOST WHEN COLLECTOR WAS OFF` metric is the total time less the covered time.

The formats for the three times mentioned are as follows:

ddd/hh:mm:ss

where *ddd* are days and *hh:mm:ss* are hours, minutes, and seconds.

Within Syntax Statements

In the previous example, the total time was 108 days, 16 hours, 14 minutes, and 51 seconds.

The number of times SCOPE was started is equal to the number of times SCOPE was restarted *or* the UTILITY command "SCOPE NEWPARM" was issued, *plus one*.

Log File Contents Summary

This summary is printed *if any* valid data was scanned. It includes the log file space and the dates covered.

```
            ------TOTAL------  --EACH FULL DAY-- -------DATES-------   FULL
TYPE        RECORDS MEGABYTES  RECORDS MEGABYTES  START       FINISH   DAYS
GLOBAL        29864     10.32   274.8      0.095 01/03/90 to 05/11/90  108.7
(NOTE RECS)     187      0.01     1.7      0.000
APPLICATION  113017     11.74  1040.0      0.108 01/03/90 to 05/11/90  108.7
PROCESS      138762     17.77  1300.1      0.166 01/22/90 to 05/11/90  106.7
DISC SPACE      214      0.06     1.0   0.000306 08/11/89 to 05/10/90  210.0
OVERHEAD                 0.73
            ------- ---------  ------- ---------
TOTAL        282044     40.63  2617.7      0.370
```

The columns can be explained as follows:

Column	**Explanation**
`TYPE`	The general type of data being logged. Two special types exist: ■ NOTES is the number of NOTES generated (SCOPE notes plus USER notes). These notes are actually kept with the global data but are listed separately in this report. ■ OVERHEAD is the amount of disc space occupied (or reserved) by the log file *versus* the amount actually used by the scanned data records. If less than the entire log file was scanned, OVERHEAD will include the data records that were not scanned. If the entire file was scanned, OVERHEAD will account for any inefficiencies in blocking the data into the file *plus* any file-access support structures. It is normal for extracted log files to have a higher overhead than raw log files since they have additional support structures for quicker positioning.
`TOTAL`	The total count and disc space scanned for each type of data.
`EACH FULL DAY`	The number of records and amount of disc space used for each 24-hour period that SCOPE runs.
`DATES`	The first and last valid dates for the data records of each data type scanned.
`FULL DAYS`	The number of full (24-hour) days of data scanned for this data type. FULL DAYS may not be equal to the difference between the start and stop dates if SCOPE coverage did not equal 100 percent of the scanned time.

The TOTAL line (at the bottom of the listed data) will give you an idea of how much disc space you are using and how much data you can expect to accumulate each day.

Log File Empty Space Summary

This summary is printed for each log file scanned.

```
THE GLOBAL      FILE IS NOW  51.6% FULL WITH ROOM FOR  103 MORE FULL DAYS
THE APPLICATION FILE IS NOW  60.1% FULL WITH ROOM FOR   74 MORE FULL DAYS
THE PROCESS     FILE IS NOW  89.1% FULL WITH ROOM FOR   13 MORE FULL DAYS
THE DISC SPACE  FILE IS NOW   9.0% FULL WITH ROOM FOR 2432 MORE FULL DAYS
```

The amount of room available for more data is calculated based on the amount of unused space in the file and the scanned value for the number of megabytes of data being logged each 24-hour day (see “Log File Contents Summary”). If the megabytes-scanned-per-day values appear unrealistically low, they are replaced with default values for this calculation.

Note This report is made on a file-by-file basis. NOTES on the previous report are included with the global data. OVERHEAD, which was reported as combined overhead in the previous report, is calculated for individual files in this report.

If you scan an extracted file, you will get a single report line since all data types share the same extracted file.

Glossary

This glossary contains an alphabetized list of terms associated with Performance Collection Software.

alarm action
The command that is executed when a performance alarm is started, repeated, or finished.

alphanumeric
A character set composed of numbers, letters, or a combination of both.

application
A Performance Collection Software application is a user-defined group of related processes.

APPLICATION=
A Performance Collection Software PARM file directive that defines a collection of programs and reports on their combined activities.

application data
The data that is logged to Performance Collection Software application log files that contain measurements of processes combined into user-defined groups (applications).

application log file
The raw log file, LOGAPPL, where the SCOPE(XL) collection program places summarized measurements of the processes in each user-defined group (application).

Within Syntax Statements

ASCII
An acronym for the **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange. ASCII is a set of standard codes representing letters, numbers, and special characters. ASCII codes are used to exchange information between computers and peripheral devices.

ASCII file
A text file that contains ASCII code representing letters, numbers, and special characters. The file does not contain any special nonprinting characters.

binary file
A file that contains machine readable codes representing letters, numbers, and special characters. The file may contain special nonprinting characters.

cache (disc cache)
A facility on some HP 3000 systems that allows portions of disc contents to be maintained in the memory of the main processor.

collection
The Performance Collection Software procedure for gathering HP 3000 performance measurement data on system resource utilization, terminal transaction rates, and response times. On MPE/iX systems, SCOPE(XL) logs this data to as many as five different log files: global, application, process, disc space, and index.

CPU
An acronym for the **C**entral **P**rocessing **U**nit—the functional part of a computer that executes program instructions.

DAILYTIME
A Performance Collection Software PARM file directive that sets the time of day when the daily sampling of the disc data is done.

data file
A text file that contains ASCII code representing letters, numbers, and special characters. The file does not contain any special nonprinting characters. All non-numeric fields in this file are bounded by double quote characters. This file format is often the preferred format for importing data into many popular spreadsheet programs.

default
An option that is selected automatically by the system if it is not overridden by a user-selected option or an appropriate command.

detail data
Raw data that is collected every 5 minutes by SCOPE(XL).

empty space
The difference between the maximum possible size of a log file and its current size.

encryption/decryption
A file-security mechanism that alters access to the contents of a file by making its contents unreadable by unauthorized users.

exported log file
A file created by the Performance Collection Software EXTRACT program (EXPORT command) or HP LaserRX/MPE Analysis Software (EXPORT LOGFILE or EXPORT WINDOW commands). It contains user-selected data ranges and types of data in one of several industry standard formats. An exported file is designed for direct printing or for use by other tools and programs. It cannot be used as input to Performance Collection Software or HP LaserRX/MPE Analysis Software programs.

EXTRACT
A Performance Collection Software program that allows the user to create extracted log files containing selected performance data for specific analytical needs.

extracted log file
A performance measurement log file created by the Performance Collection Software EXTRACT program. It contains user-selected data ranges and types of data.

extraction
The Performance Collection Software procedure for selecting desired data using the HP 3000 EXTRACT program.

file equations
The MPE commands used to redirect the file attributes for programs. For example, you can use a file equation to change the name of an extracted file.

FINISH
A Performance Collection Software PARM file directive that specifies the alarm action to be taken when a performance alarm is no longer satisfied.

global
A qualifier that implies the entire system.

global data
The data, logged to Performance Collection Software global log files, that contain measurements of systemwide activity.

global log file
The raw log file, LOGGLOB, in which the collection program stores measurements of systemwide performance.

host computer
See HP 3000 server.

HP 3000 server
The HP 3000 in which the Performance Collection Software is installed to give you access to remote data. It is the *host* computer.

ICS
An acronym for the **I**nterrupt **C**ontrol **S**tack whose principal activity is to handle interrupts.

ID
A Performance Collection Software PARM file directive that identifies a host system.

IF
A Performance Collection Software PARM file directive that defines a condition that *must be true* to be able to satisfy a performance alarm.

interesting process
A process, logged into a process log file, that becomes interesting when first created or terminated, or when it exceeds certain user-defined thresholds.

intervals
The specific Performance Collection Software time periods during which SCOPE(XL) logs data.

log files
Performance Collection Software data files that are either raw or extracted. Extracted log files are optimized for quick retrieval by the PC programs and can be copied to the PC for local access.

LOG
A Performance Collection Software PARM file directive used to specify the type of data to be written to appropriate log files.

LOGAPPL
A Performance Collection Software raw log file that contains measurements of the activity in each user-defined application.

LOGDISC
The raw log file containing summarized measurements of disc space usage.

LOGGLOB
A Performance Collection Software raw log file containing measurements of the systemwide activity.

LOGINDX
The raw log file (only in MPE/iX) containing information used to rapidly index the log files when retrieving information. On MPE V systems, this information is kept in each of the other log files, so a LOGINDX file is not needed.

LOGPROC
A Performance Collection Software raw log file containing measurements of selected interesting processes.

main memory
Computer memory that is used directly by the CPU.

MAINTTIME
A Performance Collection Software PARM file directive that sets the time of day at which scheduled maintenance operations will be performed.

MAXTHINK
A THRESHOLD parameter that sets a filter used in calculating terminal response times. Any terminal transaction where the think time was greater than MAXTHINK seconds is ignored.

measurement interface
An HP 3000 performance data source used by Performance Collection Software's performance data collection facility.

memory manager
The part of the operating system that controls main and virtual memory. This activity can consume both CPU and disc I/O resources.

MINTHINK
A THRESHOLD parameter that sets a filter used in calculating terminal response times. Any terminal transaction where the think time was less than MINTHINK seconds—such as a terminal status read—is assumed to be a hardware-generated transaction.

MPE
An acronym for **M**ulti-**P**rogramming **E**xecutive—the HP 3000 operating system.

NOKILLED
A THRESHOLD parameter that prevents the logging of any process that was considered interesting *only* because it was terminating *and* was not interesting for any other reason during the interval.

NONEW
A THRESHOLD parameter that prevents the logging of any process that was considered interesting *only* because it was a new process *and* was not interesting for any other reason during the interval.

NOSHORT
A THRESHOLD parameter that prevents the logging of any process that was considered interesting *only* because it was created or terminated *and* had a run time less than or equal to the NOSHORT parameter.

parameters
The options used to modify directive and command syntax to customize performance data collection.

PARM
A Performance Collection Software file that contains parameters that customize Performance Collection Software data collection.

performance alarm
A set of conditions, based on performance metrics, that indicate a needed action on a system. Performance alarms might be processed against real-time data by the SCOPE(XL) collector or against historical log file data by the UTILITY program.

performance measurement data
The data, collected in log files, that contains measurements of the effects of user-defined and system-overhead processes on that system.

process
The execution of a program file. This execution can represent an interactive (session) or batch (job) user, or an operating system.

process control block
An internal table used by MPE to manage processes.

process data
The data, logged to a Performance Collection Software process log file, that contains measurements of selected interesting processes.

process log file
The raw log file, LOGPROC, containing summarized measurements of selected interesting processes.

QUEUE
A Performance Collection Software PARM file parameter used to select processes for an application based on their CPU execution queue.

raw log file
A file that contains summarized measurements of system data. The SCOPE(XL) program collects data into raw log files: LOGGLOB containing global measurements, LOGAPPL containing application measurements, LOGPROC containing process measurements, LOGDISC containing summarized measurements of disc space usage, and LOGINDX (MPE/iX only) containing information used to rapidly index into the log files when retrieving information.

remote log file
A Performance Collection Software file that resides on an HP 3000 under the MPE file system. This file can be a raw or extracted log file.

REPEAT
A Performance Collection Software PARM file directive that sets the minimum time before a performance alarm is allowed to repeat its action.

report file
An ASCII file created by the user to define desired data metrics and the format of exported files generated by the EXTRACT program's EXPORT command.

resizing
The changing of the maximum size of a raw log file and, optionally, removing data from raw log files to ensure that a given amount of empty space exists to hold new log records. The UTILITY program is used to resize raw log files.

response time
The time between pressing ENTER or RETURN on the keyboard and obtaining a response from the computer system. Performance Collection Software records two different response times: "First Response" occurs when the first piece of data is written to the user, and "Response to Prompt" occurs when the system is ready to accept additional input and signifies this by issuing a new prompt.

rollback
The process of deleting 25 percent of a log file's data, starting with the oldest data, to make room for new data. Rollbacks occur when the SCOPE(XL) program fills a raw log file.

RUN file
The file created by SCOPE(XL) to control the running of the collector. If the RUN file is purged, SCOPE(XL) will terminate.

RXLOG
The default file created by the EXTRACT program. This file can be accessed on the host (via remote file access) or downloaded to a PC workstation.

SCOPE(XL)
The Performance Collection Software collector program that runs on the HP 3000 under the MPE operating system. It collects data from several sources in the HP 3000, but mainly from the Measurement Interface (MI), and writes (logs) this raw performance measurement data to raw log files. Raw log files reside on an HP 3000 under the MPE file system.

SCOPE2
This is the daily disc space collection program for MPE V. It is run by the SCOPE program, as needed.

SCOPEIN
A message file used for communicating between the UTILITY and SCOPE(XL) programs.

SCOPEJOB.SCOPE.SYS
The Performance Collection Software job stream that starts SCOPE(XL).

SCOPEOUT
A message file created whenever the UTILITY program is awaiting a response from SCOPE(XL). The file is created in the LOGON group of the user that runs UTILITY and should be purged automatically whenever UTILITY terminates. If it remains after UTILITY is executed, it can be purged manually.

SCOPEXL2

This is the daily, disc-space collection program for MPE/iX. It is run by the SCOPE(XL) program, as needed.

SEPARATOR

A directive in a report file that specifies the character used to separate fields in an exported file. The report file is used by the EXTRACT program during its EXPORT function.

service level

An agreement, usually between the data processing department and computer system users, that specifies the levels of acceptable service. Performance Collection Software has instrumentation to collect data for the most popular type of service-level agreement. This agreement is based on the percentage of terminal transactions that exhibit response times at or below an agreed level.

SERVICE LEVEL

A Performance Collection Software PARM file directive that defines service levels for an application. *See* service level, above.

SLA

A Performance Collection Software PARM file directive that defines service levels for an application. *See* service level, above.

STATUS

The file created by SCOPE(XL) to record status, data inconsistencies, or errors. This file is created automatically to contain noteworthy events in the life of SCOPE(XL), such as starting, stopping normally or abnormally, and resizing a raw log file.

The file is a circular ASCII file and can be printed using the FCOPY or MPE/iX PRINT command. The file may be resized to hold more or less data, but you must be careful to preserve its characteristics (circular, record length, and other parameters).

summary data

Raw data that is summarized into one data point per hour. It can be graphed on Performance Collection Software more quickly, since fewer data records must be handled to produce a graph.

Summarization reduces the size of the global and application data to approximately one-tenth to one-twelfth of the detail data size.

swap

To transfer some or all of a process's main memory resources to or from slower mass memory (disc).

system ID

The string of characters that identifies your system.

system manager

The person responsible for the operation and maintenance of a computer system.

system resources

The parts of a computer system used to fulfill users's processing demands. The CPU, disc, and main memory are the most common system resources.

terminal emulator

A program, such as HP OfficeShare's AdvanceLink, that is run on a PC but mimics an HP 3000 terminal.

THEN

A Performance Collection Software PARM file directive for defining service levels for an application. *See also* service level.

think time

The interval between the moment a system issues a prompt and when the user presses a key that initiates the next transaction.

THRESHOLD

A Performance Collection Software PARM file directive that defines how much of a system's resources a process must use before the process becomes *interesting* enough to be logged.

throughput
The number of transactions that occur in a computer during a given period.

TODAY
A keyword used in the EXTRACT and UTILITY programs to specify a starting or stopping date based on the current calendar date.

transaction
In Performance Collection Software, a terminal transaction begins when a terminal-user's input is completed (read complete) and ends when the next input can be accepted (initiation of next read to that terminal).

UTILITY
A Performance Collection Software program used to determine the contents of log files and the amount of disc space they occupy. The program is also used to resize raw log files.

WAIT THRESHOLD
The Performance Collection Software PARM file directive that lets the user specify a process as interesting and log it if it spends too much time *waiting* for a key system resource.

wild card character
A character (such as $*$, @, ?, or #) that can be included in a file name to indicate *any* character or group of characters would be acceptable in that position in another file name.

workload
The total amount of work demanded of a computer system. This work can include interactive commands and batch programs executed against various data files over a period.

variables
Entities that can assume an assigned value or a number of different values.

virtual memory
The portion of a disc (or other storage device) that is used as an extension of the main memory and is controlled by the operating system's memory manager.