CIMS Lab, Inc.

CIMS Data Collectors

for Microsoft® Windows®

Installation and User Guide

Version 3.3

CIMS Publication Number: DC-UG-330-01

Published 04/27/04

Copyright Information

Mailing Address

CIMS Lab, Inc.
3013 Douglas Blvd., Suite 120
Roseville, CA 95661-3842

Table of Contents

3 • Operating System Data Collectors

4 • Database Data Collectors

Preface

As companies continue to integrate computer technology into their business operations, it becomes increasingly important to properly administer the IT function, particularly with respect to performance and cost.

CIMS Data Collectors enable you to collect reliable and useful data related to how your technology resources are being used. CIMS Data Collectors integrate with CIMS to enable you to view IT resource consumption within your enterprise and to fairly and accurately allocate costs.

The technology behind CIMS Data Collectors is based on CIMS Lab's many years of experience in the development and implementation of Resource Accounting, Capacity Planning, and IT Chargeback products.

About CIMS Lab

Founded in 1974, CIMS Lab has focused on meeting the financial and resource reporting requirements of Information Services Departments. CIMS has evolved with corporate IT management requirements. Focused commitment to client service and support sets CIMS apart from competing products. Our goal is to provide the best chargeback and resource reporting software in the world at the lowest possible cost to our customers.

CIMS Lab strongly believes in and executes the concept of continuous product improvement. Customers have access to CIMS product development personnel to ensure that customer feedback and other critical issues are incorporated into the next release of the product.

Contacting CIMS Lab

To contact CIMS Lab with questions, comments or problems, please use one of the following methods:

For product assistance or information:

USA & Canada, toll free - (800) 283-4267
International - (916) 783-8525
FAX - (916) 783-2090
World Wide Web - http://www.cimslab.com

Mailing Address:

CIMS Lab, Inc.
3013 Douglas Blvd., Suite 120
Roseville, CA 95661-3842

About This Guide

This guide provides instructions for installing, setting up, and running CIMS Data Collectors. Because of its technical content, this guide is primarily intended for users that have experience working with the following Microsoft technologies:

- Windows Script Host (WSH)
- Visual Basic Script (VBScript)
- Component Object Model (COM)

Although this guide refers primarily to the use of CIMS Data Collectors with CIMS Server, you can also use CIMS Data Collectors with CIMS Desktop and CIMS Mainframe Data Collector and Chargeback System. However, you will need to modify some of the procedures and files described in this guide for these CIMS products. For assistance, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

This guide assumes that users are familiar with concepts associated with the CIMS Server application including its architecture and functions and the layout and use of the CIMS Server Resource (CSR) file. For more information about CIMS Server, including the layout of the CSR file, refer to the *CIMS Server Administrator's Guide*.

How to Use this Guide

The following table describes the chapters in this guide. You should begin with *Chapter 2, Installing CIMS Data Collectors and Setting Up the System* and then continue to the collector-specific information provided in Chapter 3 through Chapter 12.

Ch. No.	Chapter Name	Content Description
1	*About CIMS Data Collectors*	Provides an introduction to CIMS Data Collectors.
2	*Installing CIMS Data Collectors and Setting Up the System*	Provides steps for installing and setting up CIMS Data Collectors and an overview of the system architecture.
3	*Operating System Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for operating systems.
4	*Database Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for databases.
5	*E-mail Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for e-mail applications.
6	*Internet Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for internet applications.
7	*Storage Data Collectors*	Provides setup procedures for CIMS Data Collectors for storage systems.
8	*Network Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for network applications.
9	*Printer Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for printers.
10	*Transactions Collector*	Provides setup procedures for the CIMS Data Collector for transactions.
11	*Other Data Collectors*	Provides logging and setup procedures for CIMS Data Collectors for miscellaneous applications.
12	*CIMS Universal Data Collector*	Provides procedures for using CIMS Universal Data Collector.
13	*Contacting Technical Support*	Provides technical support information.
A	*CIMS Aggregation Engine API*	Describes the CIMS Aggregation Engine API.
B	*CIMS Processing Engine API*	Describes the CIMS Processing Engine API.
C	*Running Batch Scripts*	Provides procedures for running the batch scripts for CIMS Data Collectors.
D	*Adding Data Sources*	Provides procedure for adding data sources to the Windows ODBC Data Source Administrator.

Document Conventions

Some or all of the following conventions appear in this guide:

Symbol or Type Style	Represents	Example
Alternate color	(online help only) hyperlinked cross-references to other sections in this guide; if you are viewing this guide online, you can click the cross-reference to jump directly to its location	.....see Data Migration.
Italic	words that are emphasized	...the entry *after* the current entry...
	a new term	...by *identifier* values.
	the titles of other manuals	*CIMS Server Administrator's Guide*
	variables in file names	`CIMSProcessLog-yyyymmdd.txt`
Bold	names of interface items such as tabs, boxes, buttons, lists, and check boxes.	Select the **Use Local Time** check box Enter the path in the **Log File Path** box
`Monospace`	directories, file names, command names, computer code, computer screen text, system responses, command line commands, what the user types	`Processes` folder `PreMSSQL2000.wsf` script Type `Nightly preday`
< >	the name of a key on the keyboard	Press <Enter>
▸	navigating a menu or a folder	File ▸ Import ▸ Object

Terminology Used in this Guide

For simplicity, in this guide, the term "application" refers to both applications and systems.

Related Publications

As you use this guide, you might find it helpful to have these additional guides available for reference:

- *CIMS Server Administrator's Guide*
- *CIMS Desktop User Guide*
- *CIMS Mainframe Data Collector and Chargeback System User Guide*
- *CIMS Data Collector for UNIX User Guide*

1

About CIMS Data Collectors

What are CIMS Data Collectors?

CIMS Data Collectors read standard usage metering files (such as log files) generated by applications and produce a common output file that integrates with the CIMS cost allocation and chargeback system. (See *How CIMS Data Collectors for Microsoft Windows Integrate with Other CIMS Components* on page 1-5).

CIMS Data Collectors are non-intrusive and do not affect system performance or operation. Most collectors gather data from files that are produced by an application's built-in usage metering functionality.

CIMS Data Collectors are available for the following platforms: Microsoft® Windows®, Unix, Linux, and Oracle, and Mainframe Systems.

This guide describes the data collectors that run on the Windows operating system. All references to CIMS Data Collectors in the following chapters refer to the collectors for the Windows system.

For information about data collection for other platforms, refer to the CIMS Chargeback for UNIX and CIMS Chargeback for MVS documentation.

About CIMS Data Collectors for Microsoft Windows

CIMS Data Collectors for Microsoft Windows run on Microsoft 2000 Server or later operating system and collect data from Windows and Windows-compatible applications (including older Windows operating systems such as Windows NT) and non-Windows applications and operating systems.

CIMS Lab provides two types of Windows data collectors:

- Application-specific collectors for commonly used Windows and Windows-compatible applications.
- A universal collector, CIMS Universal Collector, for applications that do not have a specific collector. CIMS Universal Collector allows for the collection of *any usage metering data from any application*.

The following sections list the applications that are supported by both data collector types.

CIMS Application-specific Data Collectors

CIMS Data Collectors are available for the following applications.

Operating Systems	Microsoft Windows NT 4.0, Microsoft Windows 2000/XP
Databases	Microsoft SQL Server, Oracle, DB2
Internet Applications	Microsoft Internet Information Services (IIS), Microsoft ISA/Proxy Server, and others
E-mail Applications	Microsoft Exchange Server, Microsoft Outlook Web Access
Storage Systems	Windows File Systems (NTFS) and others
Network Applications	Novell Netware
Others	Printers, SAP, and others

Table 1-1 • CIMS Data Collectors for Windows and Windows-compatible applications

CIMS Universal Data Collector

The following table lists some (not all) of the applications supported by CIMS Universal Data Collector. For information regarding applications not contained in this list, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Operating Systems	Windows, AS/400–OS/400, RS/6000–AIX, Solaris, DEC Unisys, Unix, HP/UX, Banyon-Vines, OS/2 LAN Manager, etc.
Databases	IMS, IDMS, ADABAS, Focus, Datacom, Supra, M204, Sybase, Informix, etc.
Internet/Telecom Applications	3COM Routers, CISCO Routers, RMON2, Brooks Internet, Internet Security and Acceleration Server, Surfwatch, Firewalls, Proxy Servers, Switches/Lines, PBX Systems, etc.
E-mail Applications	Lotus Notes, AOL, MSN, GroupWare, Apache, Profs, etc.
Storage Systems	Trellisoft, Northern Quota Server, CD/DVD Drives and Tape Juke Boxes, Other Storage Monitors, etc.
Network Applications	CISCO Netflow, Netscout, RMON2, SNA Server/Host Integration Server, Network Decisions, Radius, Novell Web, etc.
ERP Applications	PeopleSoft, Oracle Financials, Hyperion, JD Edwards, Lawson, BAAN, Walker, etc.
E-commerce Applications	BackWeb Sales Accelerator, Bea Systems, iPlanet EXCpert, MicroStrategy Web InfoCenter, V5 E-Business Platform, Vital Suite, B2B Commerce Platforms, Internet Resource Monitors, etc.
CRM Applications	Clarify, Pivotal, Siebel, etc.
Output Systems	SAR/Express, CA/Dispatch, BUNDL, Infopac, RMS, Control-D, Print Servers and/or RDMS Systems, Printer Accounting Server, etc.
Other Applications	BMC Patrol Suite, Candle E-Business Assurance Monitors, HP OpenView, Landmark Performance Works, Net IQ, WebTrends, etc.
Miscellaneous	NCSA Common and Extended Log File Formats, Product Specific Web Log Files, I2, Magugistics, Synquest, Paper and Forms, Supplies, Toner Delivery, Duplication Services, Microfiche, Imaging, Media, etc.

Table 1-2 • CIMS Universal Data Collector

How CIMS Data Collectors for Microsoft Windows Integrate with Other CIMS Components

CIMS Data Collectors for Microsoft Windows work with other CIMS components such as CIMS Server, CIMS Desktop, and CIMS Mainframe Data Collector and Chargeback System. CIMS uses the data provided by the CIMS Data Collectors to track usage associated with databases, software packages, in-house applications, servers and workstations, and other systems. CIMS then accurately displays the resources used and the associated charges.

As shown in Figures 1-1, it is useful to think of CIMS as a funnel that accepts usage data and returns organized information. This data is organized and restructured as a multitude of chargeback and management reports that can help IT managers and staff to track and allocate resources.

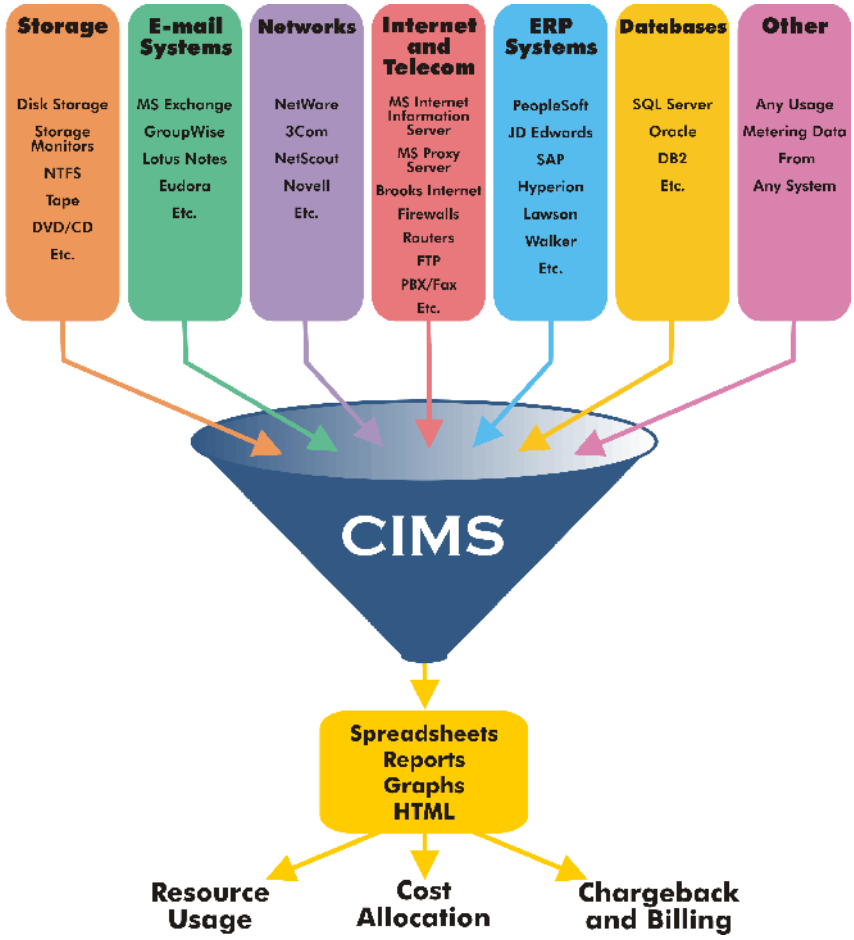

Figure 1-1 • CIMS collects usage data and organizes it as reporting information

2

Installing CIMS Data Collectors and Setting Up the System

This chapter provides the installation and configuration instructions for CIMS Data Collectors. You should review this chapter before continuing to the collector-specific chapters in this guide.

System Specifications

The following are the suggested specifications for *running* CIMS Data Collectors. Note that you can use CIMS Data Collectors to process usage metering files collected from *any* application or operating system.

- Microsoft Windows 2000 Server or Windows Server 2003 with the latest service pack. (If you are using Windows NT, contact CIMS Lab for assistance [see *Chapter 13, Contacting Technical Support*].)
- Microsoft Windows Script Host (WSH) 5.1 or 5.6 (preferred). You can download both versions free-of-charge at http://www.microsoft.com. WSH 5.1 is standard with Windows 2000 Server and will be upgraded to 5.6 if you upgrade to Microsoft Internet Explorer 6 Service Pack 1. WSH 5.6 is standard with Windows Server 2003.

Installing CIMS Data Collectors

The CIMS Server setup program also installs CIMS Data Collectors. When you install CIMS Server, you can choose to install all or certain CIMS Data Collectors (refer to the *CIMS Server Administrator's Guide* for the installation procedures).

You must install CIMS Data Collectors on the central CIMS Server server. Installation on the server enables you to process the CIMS Server Resource (CSR) files generated by the collectors through CIMS Processing Engine (also included in the CIMS Server installation).

You can also install individual collectors computers other than the central CIMS Server server in the following situations.

- You want to use the CIMS Windows Process or CIMS Windows Print collector, which produce log files containing operating system and print data, on another computer.

  CIMS Lab provides simple setup programs for installing the CIMS Windows Process and Print collectors on other computers. These setup programs install the executable and administrative programs and processing script for the collector and CIMS Aggregation Engine. For installation and setup procedures for these collectors, see *Chapter 3, Operating System Data Collectors* and *Chapter 9, Printer Data Collectors*.

- You want to convert usage metering files produced by an application to CSR files on the computer that generated the usage metering files. In most cases, this conversion is performed on the central CIMS Server server.

  To perform this conversion, you need to install the processing script for the collector and CIMS Aggregation Engine on the computer. Contact CIMS Lab (see *Chapter 13, Contacting Technical Support*) for assistance.

Setting Up the System (System Architecture)

CIMS Data Collectors are composed of the following components:

- Collectors that collect and convert the data from usage metering files to produce CSR files. Depending on the collector, usage metering files are either produced by the collector itself or produced by an application's built-in usage metering functionality.

  Examples of collectors that create usage metering files are the CIMS Windows Process and CIMS Windows Print collectors, which produce log files. These "logger" collectors are built by CIMS Lab.

  Examples of collectors that use an application's usage metering file are the Microsoft IIS, Exchange Server, and SQL Server data collectors.

- Collectors that collect and convert data directly from the system, such as the CIMS Windows Disk Collector. These collectors do not require a usage metering file to produce CSR files.

- Processes that control the process of data collection. Processes call the collectors to convert the usage metering file and call the CIMS Processing Engine to process the data and load it into the CIMS Server database.

- Scripts that support processing tasks such running the data collectors and specifying which collectors are run.

These components are stored in folders of the same name (that is, `Collectors`, `Processes` [shipped as `Sample Processes`, see page 2-8], and `Scripts`). Each folder contains the scripts and other files needed to process usage metering files. If you installed CIMS Data Collectors in the default location, these folders are in `C:\Program Files\CIMSLab`. It might be helpful to view the folders as you read the following sections.

Modifying Files and Scripts

Important! • If you modify *any* file/script in the `Collectors` or `Scripts` folder, it is very important that you rename the file. Otherwise, the file will be overwritten when you upgrade to a new version of CIMS Data Collectors.

Although you can modify any of the files and scripts shipped with CIMS Data Collectors for your site, the following scripts always require modification:

- `Processes\`*`collectorname`*`\CIMS`*`collectorname`*`.xml`. This file is used by the CIMS Windows Event Log and CIMS Windows Disk collectors.
- `Scripts\SampleNightly.bat`
- `Scripts\SampleNightly.wsf`

Depending on the collector, you might also need to modify the job script for the collector (`Processes\`*`Collectorname`*`\Job`*`datacollectorname`*`.wsf`).

Instructions for modifying these files are included in this guide. The preceding scripts (and other scripts that are likely to require modification) also include instructions in the script.

The following sections describe the files specific to the `Collectors`, `Processes`, and `Scripts` folders. A diagram showing the relationship these files is provided on page 2-20.

Collector Architecture

The `Collectors` folder contains a subfolder for each CIMS Data Collector. Depending on the collector, each subfolder contains one or more of the files described in the following sections.

Important! • If you modify a file/script in the `Collectors` folder, you need to rename the file. Otherwise, the file will be overwritten when you upgrade to a new version of CIMS Data Collectors.

Executable Program and Configuration Application

Depending on the collector, the collector subfolder might contain an executable program, and graphical user interface (GUI) program, or both.

Processing Script

Note • Each server that you are collecting from requires a separate processing script for each type of data collected. For example, if you are collecting IIS data from multiple servers, you need a separate `MSIIS.wsf` script for each server (see Figure 2-1 on page 2-20).

The processing script, *`collectorname`*`.wsf`, contains the processing instructions for converting usage metering files into CSR files. These instructions include:

- Calling CIMS Aggregation Engine (if applicable). CIMS Aggregation Engine (`CIMSAggregation.dll`) is a Component Object Model (*COM*) object that aggregates the records within the usage metering file by *identifier* values. That is, if multiple records within a file contain the same identifier values, CIMS Aggregation Engine will produce one record that contains sum total resource values for the *rate code* or codes within these records. Aggregation reduces the amount of data that CIMS Processing Engine must process and improves processing time.

  For more information about the CIMS Aggregation Engine, see *Appendix A, CIMS Aggregation Engine API*.

- Defining the chargeback identifiers and resources that are collected from the usage metering file for input into the CSR file. (Note that this is not applicable to all collectors.)

  CIMS Lab defines the most useful identifiers and resources for each collector in the collector's processing script and pre-loads these resources as rate codes in CIMS Server. You can then modify the options for these rate codes, such as description and monetary value, for your site.

  If you want to define identifiers and/or resources other than the default values in the processing script, you need to modify the script. Note that if you want to use resources other than those defined, you also need to add the rate codes for any new resources to the CIMS Rate table as described in the *CIMS Server Administrator's Guide*.

Processing Script Parameters

The processing scripts for all collectors require the parameters shown in the following table. These parameters are passed from the job script described on page 2-9.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the usage metering file(s) was created or the transaction(s) occurred (if you are using the transaction collector). The valid values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format Note that the processing script for the transaction collector, `Transaction.wsf`, accepts only `premon` and `curmon`. This parameter is entered in the command line when running the `Nightly.bat` or `Monthly.bat` script (see *Nightly.bat and Monthly.bat* on page 2-13).
`RetentionFlag`	This parameter is for future use.

Table 2-1 • Processing Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the usage metering files that you want to process or SELF if the files are on the same server as the processing script. By default, the value Server1 is included for the Feed parameter. A subfolder with the same name as the server is automatically created in the *process definition folder* (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the usage metering files (see *Feed Subfolder* on page 2-12). **Note:** Although this subfolder is created in the Transaction process definition folder, it is not used. CSR files are placed directly within the process definition folder. This parameter is included as an identifier in the CSR file with the exception of resource files created from transactions. The identifier name is Feed and the identifier value is the server name.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.

Table 2-1 • Processing Script Parameters (Continued)

In addition to the parameters in the preceding table, other parameters are required or optional depending on the specific collector. Parameters that are collector-specific are described by collector beginning with Chapter 3.

Processes Architecture

All collectors require a job script that controls the data collection process. CIMS Lab provides sample job scripts that you can modify for your site. The job scripts are stored in process definition subfolders within the `Processes` folder.

Note that there is not a process definition folder and sample job script for every collector. For those collectors that do not have a process definition folder, you can simply copy and rename an existing folder and rename and modify the job script. For more information, see the section specific to the collector that you are using beginning with Chapter 3.

Note • The CIMS Windows Event Log and CIMS Windows Disk collectors require the Microsoft .NET Framework and use *XML* files in addition to the job script. For more information about these collectors, see Chapter 7 and Chapter 9.

About the Processes Folder

The folder `Sample Processes` is shipped with CIMS Data Collectors. When you install CIMS Data Collectors for the first time, you must rename and/or move this folder before you modify any subfolders or files. CIMS Lab recommends that you rename this folder `Processes` and move it to a location where you keep data that is backed up. However, you can give this folder any name and move it to any location. This folder is referred to as `Processes` in this guide.

Important! • The path to the `Processes` folder must be defined in the configuration settings for CIMS Server. For more information, refer to the *CIMS Server Administrator's Guide*.

Each time that you upgrade to a new release of CIMS Data Collectors, a new `Sample Processes` folder is installed. You can then copy or move any new process definition folders that you want from the `Sample Processes` folder to the `Processes` folder. Each process definition folder contains the files and subfolders described in the following sections.

Job Script

The job script, Job*collectorname*.wsf, contains the processing instructions for the CIMS Data Collector.

Job Script Parameter

The job script requires the LogDate parameter—the date the usage metering file(s) was created or the transaction(s) occurred (if you are using the transaction collector). The valid values are:

- preday (previous day)
- premon (previous month)
- rndate (current day)
- curday (current day and previous day)
- curmon (current month)
- date in yyyymmdd format

Note that the job script for the transaction collector, JobTransaction.wsf, accepts only premon and curmon.

This parameter is entered in the command line when running the Nightly.bat or Monthly.bat script (see *Nightly.bat and Monthly.bat* on page 2-13).

Modifying the Job Script

The job script calls and passes parameters to the collector's processing script (if applicable) and the Scan.wsf, ProcCIMS.wsf, and CleanUp.wsf scripts (see Figure 2-1 on page 2-20). You need to modify these parameters for your site as needed.

Table 2-2 on page 2-10 provides a description of the parameters that are included in the job scripts for *all* collectors. In addition to the parameters shown in this table, the job script might include collector-specific parameters. For a description of these parameters, see the section specific to the collector that you are using beginning with Chapter 3.

You may also need to add optional parameters for use by other scripts in the collector architecture. For a description optional parameters by script, see the *Scripts Architecture* section beginning on page 2-13

Note • The CIMS Windows Event Log and CIMS Windows Disk collectors include an XML file that enables you to modify parameters without modifying the job script. For more information about these collectors, see Chapter 7 and Chapter 9.

Parameter	Description/Values
Located under `Initial Settings`	
`DataSourceID`	The ODBC data source or data source UID for the CIMS Server database. If you are using only one database with CIMS Server, leave this parameter value blank. The data source selected in the Select ODBC Data Source dialog in CIMS Server Administrator box is used as the data source. If you are using multiple databases with CIMS Server, enter the UID for the data source. The data source UID is entered in the Data Source List Maintenance dialog box in CIMS Server Administrator. For more information about creating and using data sources, refer to the *CIMS Server Administrator's Guide*.
`ShortProcessFolder`	The process definition folder for the collector. This parameter should not be changed unless you changed the name of the process definition folder or you copied and modified the folder for use by another collector. This is the location of the final CSR file that is created by the `Scan.wsf` script (see page 2-16).
`CollectorName`	The collector name. This is the same as the collector folder name. This parameter should not be changed.
`ProcessesFolder`	The full path of the `Processes` folder. This parameter should not be changed.
`CIMSFolder`	The full path of the CIMS installation. This parameter should not be changed.
`CollectorsFolder`	The full path of the `Collector` folder. This parameter should not be changed.
`RetentionFlag`	This parameter is for future use.
`DaysToRetainFiles`	Specifies when CSR files are to be deleted from the process definition folder. For more information about this parameter, see page 2-18.
`CleanSubfolders`	Specifies whether the CSR files that are contained in the server subfolders are also deleted. For more information about this parameter, see page 2-18.

Table 2-2 • Parameters Passed from the Job Script

Parameter	Description/Values
Located under `TODO Run Preprocess to format...`	
`Feed`	The name of the server that contains the usage metering files that you want to process or `SELF` if the files are on the same server as the processing script. By default, the value `Server1` is included for the `Feed` parameter. A subfolder with the same name as the server is automatically created in the process definition folder (see the `ShortProcessFolder` parameter). This subfolder is used to store the CSR files that are created from the usage metering files (see *Feed Subfolder* on page 2-12). **Note:** Although this subfolder is created in the `Transaction` process definition folder, it is not used. CSR files are placed directly within the process definition folder. This parameter is included as an identifier in the CSR file with the exception of resource files created from transactions. The identifier name is `Feed` and the identifier value is the server name.

Table 2-2 • Parameters Passed from the Job Script (Continued)

If you are collecting from multiple servers, copy and edit the following to add steps for each server (feed) that you are collecting from. If you are collecting from only one server, comment the second `Call RunCollection(LogFolder,Feed)`.

This code example is from the `JobMSIIS-Web.wsf` script.

```
' ----------------------------------------------------------------------
    ' TODO Run Preprocess to format IIS logs
    ' ----------------------------------------------------------------------

        IISProcessType = "Web"

        ' ------------------------------------------------------------------------
        ' Beginning of first server
        ' ------------------------------------------------------------------------
        LogFolder               = "\\Server1\c$\Winnt\system32\LogFiles"
                                   'This is the location of the output Log files from IIS
        Feed                    = "SERVER1"
        Call RunCollection(LogFolder,Feed)


        ' ------------------------------------------------------------------------
        ' Beginning of second server
        ' ------------------------------------------------------------------------
        LogFolder               = "\\Server2\c$\Winnt\system32\LogFiles"
                                   'This is the location of the output Log files from IIS
        Feed                    = "SERVER2"
        Call RunCollection(LogFolder,Feed)
```

XML File

The CIMS Windows Event Log and CIMS Windows Disk collectors include an XML file, `CIMS`*`collectorname`*`.xml`, that enables you to modify parameters without modifying the job script. For more information about these collectors and the XML files, see Chapter 7 and Chapter 9.

Feed Subfolder

A feed subfolder is automatically created in the process definition folder for each server that you entered as a `Feed` parameter in the job script. If you left the `Feed` parameter blank, the feed subfolder is named `Server1`.

Note • For the CIMS Windows Disk collector, a value is required for the `Feed` parameter (i.e., you cannot leave this parameter blank). For more information about this collector, see Chapter 7.

Each feed subfolder is used to store CSR files from the feed of the same name. The filename contains a date in `yyyymmdd` format. Note that although the feed subfolder is created in the `Transaction` process definition folder, it is not used. CSR files created by this collector are placed directly in the process definition folder. For more information, see Chapter 10.

The `Scan.wsf` script processes and concatenates the CSR files in the feed subfolders by the date in the filename. The resulting output file is placed directly in the process definition folder. For more information, see *Scan.wsf* on page 2-16).

Important! • To prevent data processing errors, the process definition folder should not contain subfolders other than feed folders and feed folders should not contain files other than CSR files.

Additional Processing Files

Each process definition folder contains additional processing files that are used internally by CIMS Data Collectors.

Scripts Architecture

The Scripts folder contains the scripts described in the following sections. The scripts are ordered by their position in the processing cycle.

Important! • If you modify a file/script in the Scripts folder, you need to rename the file. Otherwise, the file will be overwritten when you upgrade to a new version of CIMS Data Collectors.

Nightly.bat and Monthly.bat

Note • These scripts are shipped as SampleNightly.bat and SampleMonthly.bat. You need to rename these scripts Nightly.bat and Monthly.bat (or choose other names) so that the files are not overwritten when you upgrade to a new version of CIMS Data Collectors. These scripts are referred to as Nightly.bat and Monthly.bat in this guide.

Nightly.bat is the script that runs all CIMS Data Collectors other than the Transactions collector, which is run by Monthly.bat. You can run these scripts from the command prompt or you can use Windows Scheduled Tasks to schedule the scripts to run automatically (see *Appendix C, Running Batch Scripts*).

The Nightly.bat and Monthly.bat scripts perform the following tasks:

- Call the Nightly.wsf/Monthly.wsf script to begin processing.
- Call the SendMail.wsf script to send notification of successful, successful with warnings, or failed processing.

Batch Script Parameter

The batch script requires the LogDate parameter—the date the usage metering file(s) was created or the transaction is to occur. The valid values for each batch script are:

Nightly.bat:

- preday (previous day, this is the most commonly used parameter for Nightly.bat)
- premon (previous month)
- rndate (current day)
- curday (current day and previous day)
- curmon (current month)
- date in yyyymmdd format

Monthly.bat:

- premon (previous month)
- curmon (current month)

This parameter is entered in the command line. For example, if you are running the Nightly.bat script from Scheduled Tasks, the command for processing usage metering files produced on the previous day is "C:\Program Files\CIMSLab\Scripts\Nightly.bat" preday (if you installed CIMS Data Collectors in the default location).

Modifying the Batch Script

You need to modify the batch script as follows:

1. Change the call to `SampleNightly.wsf` or `SampleMonthly.wsf` to `Nightly.wsf`/`Monthly.wsf` (or choose other names). Make sure you rename the actual `SampleNightly.wsf`/`SampleMonthly.wsf` scripts accordingly (see *Nightly.wsf and Monthly.wsf* on page 2-15).
2. Change the following for your organization:
   - The Simple Mail Transfer Protocol (SMTP) server (the default is `mail.cimslab.com`).
   - The sender address for the collector processing results e-mail message (the default is `cims.server@cimslab.com`).
   - The receiver address for the collector processing results e-mail message (the default is `CIMSProcessResults@cimslab.com`). If you want to use multiple addresses, separate them with a semicolon (;).
   - The location of the `Nightly.txt` or `Monthly.txt` file if used. (For a description of these files, see *SendMail.wsf* on page 2-19.) These files are located in the `Scripts` folder by default. If you move these files, the full path is required. The use of a UNC for the path is recommended.

You can also change the subject of the collector processing results e-mail. The defaults are:

```
:ErrorHandler
Set OUTMSG="CIMS Ended with a completion code of %ERRORLEVEL%"
GOTO Exit

:Warning
Set OUTMSG="CIMS Ran Successfully with a completion code of %ERRORLEVEL%"
GOTO Exit

:SeriousWarning
Set OUTMSG="CIMS Ended with a completion code of %ERRORLEVEL%"
GOTO Exit

:Success
Set OUTMSG="CIMS Ran Successfully with a completion code of %ERRORLEVEL%"
```

Where `%ERRORLEVEL%` represents one of the following return codes:

`0`	Execution ended with no errors or warnings.
`4` or `8`	Execution ended with warning messages.
`16`	Execution ended with errors—processing stopped.

Nightly.wsf and Monthly.wsf

Note • These scripts are shipped as `SampleNightly.wsf` and `SampleMonthly.wsf`. You need to rename these scripts `Nightly.wsf` and `Monthly.wsf` (or choose other names) so that the files are not overwritten when you upgrade to a new version of CIMS Data Collectors. These scripts are referred to as `Nightly.wsf` and `Monthly.wsf` in this guide.

The `Nightly.wsf` script calls all jobs that you want to run with the exception of the transactions job, which is called by `Monthly.wsf`. These scripts contains a list of default jobs. To run a job, remove the comment. Jobs are called in the order that they appear in the list. You can add and remove jobs to and from this list.

Nightly.wsf/Monthly.wsf Script Parameter

The `Nightly.wsf/Monthly` script uses the parameters shown in the following table. These parameters are passed from the job script described on page 2-9.

Parameter	Description/Values
`LogDate`	The date the usage metering file(s) was created or the transaction is to occur (see the values for this parameter on page 2-13).
`DataSourceID (optional)`	The ODBC data source UID for the CIMS Server database (see page 2-10 for a description of this parameter).

Table 2-3 • Nightly Script Parameters

Scan.wsf

The `Scan.wsf` script performs the following tasks:

- Verifies that the feed subfolder or subfolders in a process definition folder contain a CSR file with a date in its filename that matches the `LogDate` parameter. If a matching file is not found, notification is included in the `Nightly.txt` file (see *SendMail.wsf* on page 2-19).
- Concatenates the CSR files produced by CIMS Data Collectors of the same type from multiple servers into one file.
- Outputs a CSR file (whether from one server or a concatenated file from multiple servers) to the collector's process definition folder. The default filename for the CSR file is `CurrentCSR.txt`.

Important! • If you are collecting from only one server, the use of the `Scan.wsf` script is optional. However, if you do not use the Scan script, you need to move the CSR file contained in the feed subfolder to the collector's process definition folder.

Scan Script Parameters

The `Scan.wsf` script uses the parameters shown in the following table. These parameters are passed from the job script described on page 2-9.

Parameter	Description/Values
`ProcessFolder`	The location of the CSR file created by the `Scan.wsf` script. This is the final resource file that is processed by CIMS Processing Engine.
`LogDate`	The date the usage metering file(s) was created (see the values for this parameter on page 2-13).
`RetainDateFlag (optional)`	Specifies whether the date is retained in the filename of the CSR file created by the `Scan.wsf` script. If the parameter is set to `true`, the filename is *`yyyymmdd`*`.txt`. If the parameter is not included or is set to `false`, the filename is `CurrentCSR.txt`. To retain individual CSR files by date, CIMS Lab recommends that you do no set the filename to `CurrentCSR`.
`AllowEmptyFiles (optional)`	If this parameter is set to `true`, is left blank, or is not included, a warning occurs when a blank CSR file is encountered. If this parameter is set to `false`, an error occurs and processing fails.

Table 2-4 • Scan Script Parameters

ProcCIMS.wsf

The `ProcCIMS.wsf` script calls the following CIMS Processing Engine components. These components are COM objects that process the CSR file and load the output data into the CIMS Server database.

- `CIMSACCT.dll` performs account code conversion, shift determination, date selection, and identifier extraction on the usage data and produces the CIMSACCT Detail File containing records that are properly formatted for input into CIMSBILL.
- `CIMSSORT.dll` converts the CIMSACCT Detail File and produces a sorted version that is ready to be processed by CIMSBILL. CIMSSORT also combines multiple files into one file—you can use CIMSSORT to combine CIMSACCT output from different data collectors into one file.
- `CIMSBILL.dll` processes the sorted CIMSACCT Detail File from CIMSSORT and performs shift processing, CPU normalization, proration, and include/exclude processing and creates the CIMSBILL Detail and CIMS Summary Files that contain the billing information used to generate invoices and reports.
- `CIMSAdminLib.dll` contains the class `CCIMSCBSLoad` that loads the output files from CIMSACCT and CIMSBILL into the database.

For more information about the CIMS Processing Engine COM objects, refer to *Appendix B, CIMS Processing Engine API*.

ProcCIMS Script Parameters

The `ProcCIMS.wsf` script uses the parameters shown in the following table. These parameters are passed from the job script described on page 2-9.

Parameter	Description/Values
`ProcessesFolder`	The location of the final CSR file.
`InputFileName`	The name of the final CSR file.
`LogDate`	The date the usage metering file(s) was created (see the values for this parameter on page 2-13).
`DataSourceID (optional)`	The ODBC data source UID for the CIMS Server database (see page 2-10 for a description of this parameter).
`ODBCUserID and ODBCPassword (optional)`	The parameters are deprecated and are included for legacy purposes only.

Table 2-5 • ProcCIMS Script Parameters

CleanUp.wsf

The `CleanUp.wsf` script deletes files with filenames containing the date in `YYYYMMDD` format in the collector's process definition folder. You can use the `CleanUp.wsf` script to delete files after a specified number of days from the file's creation or to delete files that were created before a specified date.

CleanUp Script Parameters

The `CleanUp.wsf` script uses the parameters shown in the following table. These parameters are passed from the job script described on page 2-9.

Parameter	Description/Values
`ProcessFolder`	The folder that contains the final CSR files.
`DaysOrDate`	Specifies when CSR files are to be deleted from the collector's process definition folder. The value for this parameter can be either: ■ The number of days from a file's creation that you want to retain the file. For example, a value of `45` specifies that all files older than 45 days from the current date are deleted. ■ A date in `yyyymmdd` format. All files created before this date are deleted.
`CleanSubfolders (optional)`	Specifies whether the CSR files that are contained in the feed subfolders are also deleted. If the parameter is set to `true`, the files are removed from the subfolders. If the parameter is set to `false` or is not included, the files are not removed.

Table 2-6 • CleanUp Script Parameters

SendMail.wsf

The SendMail.wsf script generates an e-mail at completion or failure of a data collection process. This e-mail includes an optional Nightly.txt or Monthly.txt file that provides the processing results. If processing has failed, the file indicates at which point in the process the failure occurred.

SendMail Script Parameters

The SendMail.wsf script uses the parameters shown in the following table. These parameters are passed from Nightly.bat or Monthly.bat (see page 2-13).

Parameter	Description/Values
SMTPServer	The SMTP server.
Subject	The subject of the e-mail.
FromEmail	The address of the e-mail sender.
ToEmail	The address of the e-mail receiver. If you want to use multiple addresses, separate them with a semicolon (;).
AttachFileLocation (optional)	The location of the Nightly.txt or Monthly.txt files if used. These files are located in the Scripts folder by default. If you move this file, the full path is required. The use of a UNC for the path is recommended.

Table 2-7 • SendMail Script Parameters

CIMSUtils.wsc

The Windows Script Component file CIMSUtils.wsc provides useful utilities and tools including methods for getting the Processes and CIMSLab folders and for building the Open Database Connectivity (*ODBC*) connection string.

Note • By default, the script component files CIMSUtils.wsc and Shell.wsc are registered on your computer as COM objects at installation. However, if you move a file to another computer, you need to register the file.

Shell.wsc

The Shell.wsc file is used to capture messages passed from one script to another.

CIMSLIB.wsf

CIMSLIB.wsf is a library of Windows Scripting Functions that can be used in the script files. This file is used as an include file.

CIMS Data Collectors System Architecture Diagram

Figures 2-1 shows the relationship of the components for most CIMS Data Collectors. The job and processing scripts for IIS and Microsoft SQL Server 2000 are shown for example purposes.

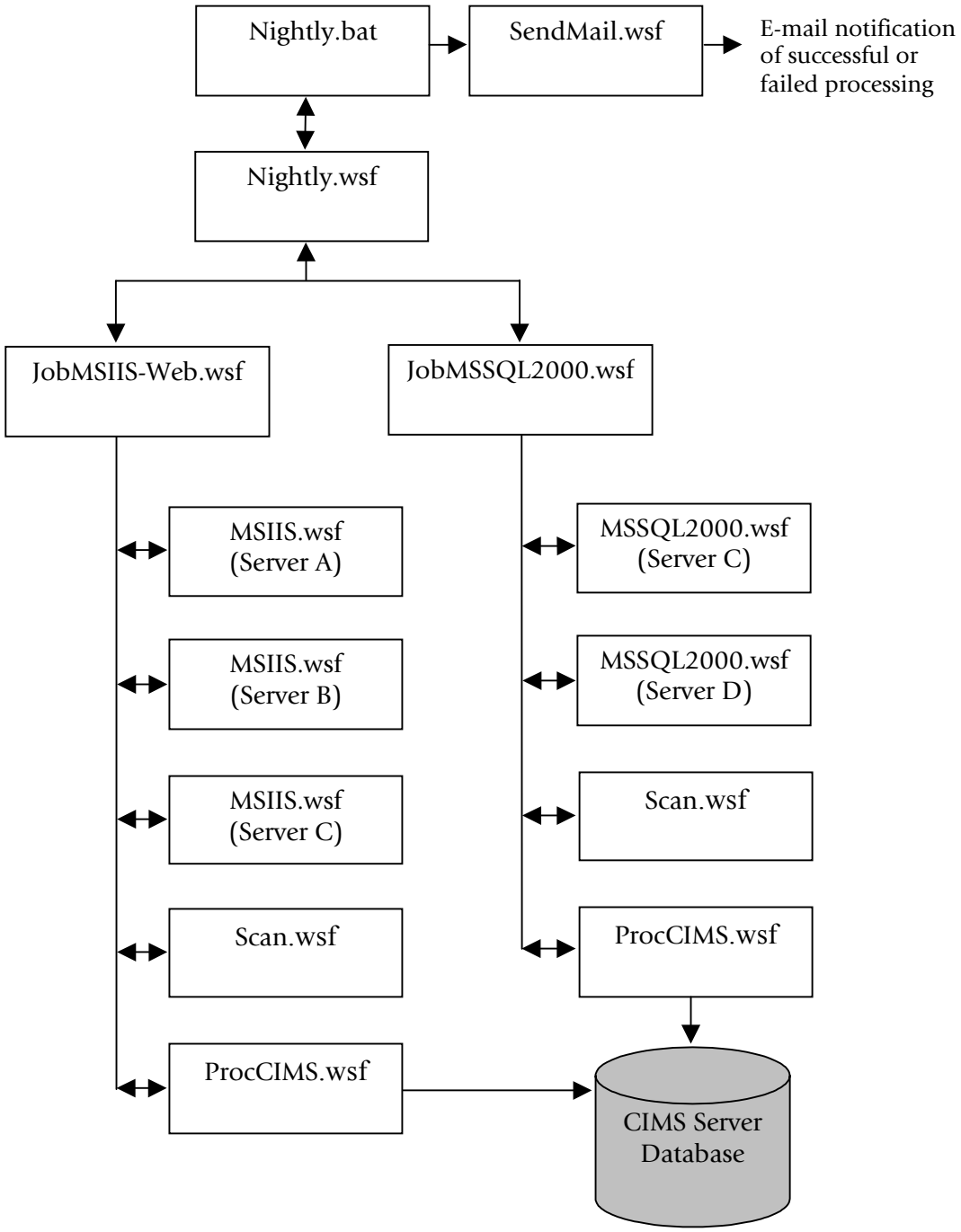

Figure 2-1 • System Architecture for CIMS Data Collectors

3

Operating System Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for operating systems. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Windows Processes Data Collector

The CIMS Windows Process collector gathers usage data for processes running on Windows 2000/2003, XP, and NT operating systems and produces a log file of the data (see *CIMS Windows Process Collector Log File Format* on page 3-6). This log file provides useful metrics such as:

- Name and path of the process.
- Name of the computer that the process ran on.
- Name of the user that created the process.
- The elapsed CPU time used by the process (cumulative and broken down by kernel and user time).
- Bytes read and written by the process.

The following sections provide instructions for installing the CIMS Windows Process collector, enabling logging, and setting up and running the collector.

Installing the CIMS Windows Process Collector

In most cases, you will want to collect process usage data from computers other than the central CIMS Data Collectors server. CIMS Lab provides a simple setup program, `CIMSWinProcessSetup.exe`, for installing the CIMS Windows Process collector on other computers. This setup program includes the following components:

- **The CIMS Windows Process collector.** This installs the following components in the `Collectors\CIMSWinProcess` folder created during installation:
  - The processing script, `CIMSWinProcess.wsf`.
  - The executable program for the collector, `CIMSWinPService.exe`.
  - An executable program, `CIMSWinPServiceLog.exe`, that is used by CIMS Lab for troubleshooting purposes. For more information about this program, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).
  - The executable program for the collector's administrative program, `CIMSWinPServiceAdmin.exe`.
- **CIMS Aggregation Engine (`CIMSAggregation.dll`).** CIMS Aggregation Engine is called by the `CIMSWinProcess.wsf` script. CIMS Aggregation Engine aggregates the records within process log file by identifier values and produces a CIMS Server Resource (CSR) file. For more information about CIMS Aggregation Engine, see *Processing Script* **on page 2-5**.
- **Support Files.** These files support the collector's administrative program and are needed only if **CIMS Server Administrator** is not installed on the computer.

This installation does not include CIMS Processing Engine, which processes the CSR files created by CIMS Aggregation Engine and loads the output data into the database. To process CSR files, you need to process the files on the central CIMS Data Collectors server. For more information, see *Setting Up the CIMS Windows Process Collection Process* on page 3-11.

To install the CIMS Windows Process collector:

Note • These following steps are also applicable if you are upgrading to a new version or release of the CIMS Windows Process collector.

1. Log on to Windows as an Administrator.
2. Click the Windows **Start** button, and then click **Run**.
3. Enter the path to the setup program `CIMSWinProcessSetup.exe`, and then click **OK**. (The path depends on the location of the setup program, i.e., CIMS Lab Product CD, network drive, etc. For example, if you are installing from CD, the path might be `D:\CIMSServer\CIMSWinProcessSetup.exe`).
4. In the setup wizard, select the collector components or component that you want to install as follows, and then click **Next**:
   - To install multiple components, click **Custom** and then select the check boxes for the components that you want to install. Make sure that you select **CIMS Aggregation Engine** and **Support Files** if these components are not already installed on the computer. (For a description of these components, see page 3-2.)
   - To install an individual component, click the component that you want to install.
5. Choose the default location for installation (`C:\Program Files\CIMSLab`) or click **Browse** to choose another location. After making your selection, click **Install**.

Enabling CIMS Windows Process Logging

To enable logging for the CIMS Windows Process collector, you need to create an output folder for the log files created by the collector and to configure the collector as described in the following sections.

Creating an Output Folder for Storing Log Files

You need to create an output folder for storing the log files that are created by the CIMS Windows Process collector. This folder should be on the same computer as the collector, not on the central CIMS Data Collectors server. You should create this folder in a location where you keep data that is backed up.

Important! • Do not store log files (`CIMSProcessLog-`*`yyyymmdd`*`.txt`) in the `Processes\CIMSWinProcess\`*`feedname`* folder on the central CIMS Data Collectors server. The feed folder should contain only CSR files.

Configuring the CIMS Windows Process Collector

The CIMS Windows Process collector runs at configurable intervals and tracks all processes that are running at that time until the completion of the process. The usage data for each process is entered as a record or records in the log file.

The CIMS Windows Process collector includes an easy-to-use GUI administrative program for configuring and enabling the collector. To use this program, click the **Start** menu, and then click **Programs ▸ CIMS Server ▸ Collectors ▸ CIMS Windows Process Administrator—Service** and set the following options:

- **Log file path.** Enter the path to the folder that you created in *Creating an Output Folder for Storing Log Files*. The default is the path where the `CIMSWinPService.exe` program is located. The use of a UNC path for the file location is recommended.
- **Log file prefix.** The default name for the log file is `CIMSProcessLog-`*`yyyymmdd`*`.txt`. You can use the default prefix `CIMSProcessLog-` or replace it with the prefix of your choice (or no prefix).
- **Use Local Time in output records.** If this check box is selected (the default), the local time set for the computer is used in the date and time fields in the log file. If this check box is cleared, Universal Time Coordinate (*UTC*) time is used in the log file.

Note • The log file *date* always reflects local time, regardless of whether Use Local Time is selected.

- **Sampling**

  - **Look for new process every.** Enter the number of seconds, minutes, or hours that you want to begin tracking new processes. For example, if you set the sampling interval to 5 seconds, the collector checks every 5 seconds to determine which new processes have began since the last check and tracks those processes until completion.

    You can use the sampling option alone or in conjunction with the interval accounting option. If you do not select the **Enable Interval Accounting** check box, a cumulative End record is created in the log file when the process ends. (For a description of the types of records that are contained in the log file, see *CIMS Windows Process Collector Log File Format* on page 3-6.)

    Note • The CIMS Windows Process **collector does not collect data for processes that run between sampling intervals.**

- **Accounting**

  - **Enable Interval Accounting.** Select this check box to use interval accounting.

    The use of interval accounting is recommended for chargeback because it provides Start, Interval, and optional End records for a process rather that just a final End record. This is especially beneficial for long running processes that begin in one billing period and end in another.

    When you select interval accounting, a Start record is created in the log file when the Windows collector begins tracking the process. Interval records are created at the interval times that you set in the **Write accounting records every** boxes until the process ends. If you select the **Write End records** check box, an End record containing a cumulative total for the process is also created.

  - **Write accounting records every.** Enter the number of seconds, minutes, or hours that you want to create interval records. For example, if you set interval accounting to every 15 minutes the following records are produced:

    - A Start record with an elapsed time showing the amount of time in seconds that the process had been running when the collector began to track it. For example, if the process had been running for 2 minutes, the elapsed time for the Start record is 120.

    - An Interval record with an elapsed time of 900 for each 15 minute interval that occurs during the process. If the process ends before 15 minutes, an interval record is created showing the time that the interval ran. Likewise, if the process ends between 15 minute intervals, a final interval record is created showing the time that the interval ran.

- **Write End records.** Select this check box if you want End records to be included in the log file in addition to Start and Interval records. Because the End record provides cumulative totals of the usage totals shown in the Start and Interval records, you may not want to include End records when using interval accounting. For chargeback purposes, the resulting total usage amounts from the combined Start, Interval, and End records will be double the actual usage amount if the amounts are not filtered by the `CIMSWinProcess.wsf` script.

- **Control Service.** Click this button open the Service Control dialog box to start or stop the CIMS Windows Process collector. You can also start and stop the collector from the Windows Control Panel and then click **Refresh Status** in the Service Control dialog box to make the change in the collector.

CIMS Windows Process Collector Log File Format

The following table describes the record fields in the log file produced by the CIMS Windows Process collector.

There are three types of records that might appear in the log file:

- Start records, which provide usage data for the start of a process.
- Interval records, which provide usage data for the intervals between the start and end of a process.
- End records, which provide summary usage data at the end of a process. All totals in an End record are cumulative for the whole process.

Start and Interval records appear only if the collector is configured for interval accounting.

End records appear in the following situations:

- If the collector *is not* configured for interval accounting. In this situation, only End records appear.
- If the **Write End records** check box is selected for interval accounting (see page 3-6).

Note • The term "process" in the following table can refer to the entire process, or the start, interval, or end of a process depending on whether interval accounting is used.

Field Name	Description/Values
Record Type	S = Start of process (note that this does not appear if interval accounting is not used) I = Interval (note that this does not appear if interval accounting is not used) E = End of process (this record appears if you do not enable interval accounting or if you enable interval accounting and select the **Write End records** check box)
ProcessID	Process identifier (PID) assigned to the process by the operating system.
ParentProcessID	The PID for the entity that created the process. Assigned by the operating system.
ProcessName	The name of the process.
ProcessPath	The path where the process executable is located.
MachineName	The name of the computer running the process.
UserName	The name of the user that created the process.
TerminalServicesSessionID	If Microsoft Terminal Services is used to access the process on the computer, the session ID.
CreateDateTime	The date and time that the process was created.
ExitDateTime	The date and time that the entire process ended.
ExitCode	The result from the completion of the process.
IntervalStartDateTime	If using interval accounting, the date and time that the interval started.
IntervalEndDateTime	If using interval accounting, the date and time that the interval ended.
ElapsedTimeSecs	The total elapsed time in seconds for the process.
CPUTimeSecs	The total elapsed CPU time in seconds for the process. This field is the sum of KernelCPUTimeSecs and the UserCPUTimeSecs fields.
KernelCPUTimeSecs	The total elapsed time in seconds that the process spent in kernel mode. (For a description of kernel mode, see *About Kernel Mode and User Mode* on page 3-9).

Table 3-1 • CIMS Windows Process Collector Log File Format

Field Name	Description/Values
UserCPUTimeSecs	The total elapsed time in seconds that the process spent in user mode. (For a description of user mode, see *About Kernel Mode and User Mode* on page 3-9).
Read Requests	The number of read requests made by the process.
KBytesRead	The number of kilobytes read by the process.
Write Requests	The number of write requests made by the process.
KBytesWritten	The number of kilobytes written by the process.
PageFaultCount	In a paged virtual memory system, an access to a page (block) of memory that is not currently mapped to physical memory. When a page fault occurs, the operating system either fetches the page from secondary storage (usually disk) if the access is legitimate or reports the access as illegal if access is not legitimate. A large number of page faults lowers performance.
WorkingSetSizeKB	The amount of memory in kilobytes mapped into the process context.
PeakWorkingSetSizeKB	The maximum amount of memory in kilobytes mapped into the process context at a given time.
PagefileUsageKB	The amount of memory in kilobytes that is set aside in the system swapfile for the process. It represents how much memory has been committed by the process.
PeakPagefileUsageKB	The maximum amount of memory in kilobytes that is set aside in the system swapfile for the process.
PriorityClass	The priority class for the process. Assigned by the operating system.
BasePriority	The priority with which the process was created. Assigned by the operating system.
SystemProcessorCount	The number of processors on the computer.
EligibileProcessorCount	The number processors on the computer that the process is allowed to use.
AffinityMask	A bit mask value indicating which processors the process may run on.

Table 3-1 • CIMS Windows Process Collector Log File Format (Continued)

About Kernel Mode and User Mode

The kernel mode is where the computer operates with critical data structures, direct hardware (IN/OUT or memory mapped), direct memory, interrupt requests (IRQs), direct memory access (DMA), etc.

The user mode is where users can run applications. The kernel mode prevents the user mode from damaging the system and its features.

Figure 3-1 shows the relationship of the kernel and user mode.

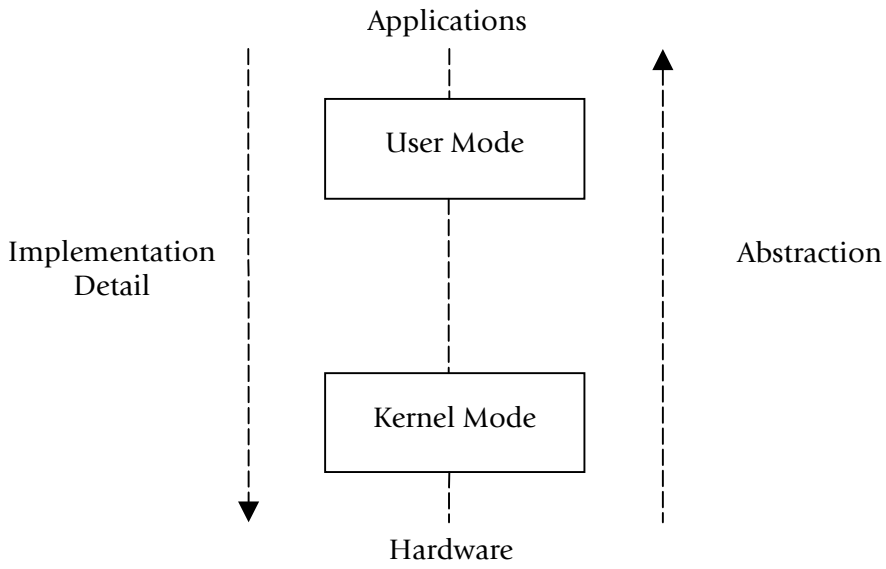

Figure 3-1 • Kernel and User Mode

Identifiers and Resources Collected from the CIMS Windows Process Collector Log File

By default, the following fields in the CIMS Windows Process collector log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the CIMS Windows Process collector processing script, `CIMSWinProcess.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
—	Feed (passed from the `JobCIMSWinProcess.wsf` script)	—
`ProcessName`	ProcessName	—
`ProcessPath`	ProcessPath	—
`MachineName`	Server	—
`UserName`	User	—
`PriorityClass`	PriorityClass	—
`BasePriority`	BasePriority	—
Resources		
`ElapsedTimeSecs`	MS Windows Elapsed Time	WINELPTM
`CPUTimeSecs`	MS Windows CPU Time	WINCPUTM
`KernelCPUTimeSecs`	MS Windows Kernel CPU Time	WINKCPUT
`UserCPUTimeSecs`	MS Windows User CPU Time	WINCPUUS
`Read Requests`	MS Windows Read Requests	WINRDREQ
`KBytesRead`	MS Windows KB Read	WINKBYTR
`Write Requests`	MS Windows Write Requests	WINWRREQ
`KBytesWritten`	MS Windows KB Written	WINKBWRI
`PageFaultCount`	MS Windows Page Fault Count	WINPGFLT

Table 3-2 • Default CIMS Windows Process Identifiers and Resources

Setting Up the CIMS Windows Process Collection Process

The following sections provide steps for setting up the CIMS Windows Process collection process. These steps differ depending on whether you are processing the log files produced by the CIMS Windows Process collector on the central CIMS Data Collectors server or whether you are processing the log files on the computer running the CIMS Windows Process collector.

Note • Although you can process log files on the computer running the CIMS Windows Process collector, you should not process the resulting CSR files on this computer. You should process CSR files on the CIMS Data Collectors server.

Of the two options for processing log files, processing the log files on the central CIMS Data Collectors server is the simpler option. However, if the log files are large, you should have a quicker elapsed completion time if you process the files on the computer running the CIMS Windows Process collector.

Option 1—To process the log files on the central CIMS Data Collectors server:

On the central CIMS Data Collectors server, set up the collector and related scripts as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The job script for the collector is `JobCIMSWinProcess.wsf`. The processing script for the collector is `CIMSWinProcess.wsf`.

If you installed the CIMS Windows Process collector in the default location on the central CIMS Data Collectors server, the `CIMSWinProcess.wsf` script is in `C:\Program Files\CIMSLab\Collectors\CIMSWinProcess`. The `JobCIMSWinProcess.wsf` script is in `Processes\CIMSWinProcess` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

The `JobCIMSWinProcess.wsf` script passes parameters to the `CIMSWinProcess.wsf` script. These parameters are described in *Setting the Parameters for the CIMSWinProcess.wsf Script* on page 3-12.

Option 2—To process the log files on a computer running the CIMS Windows Process collector:

1. On the computer running the CIMS Windows Process collector, create a batch script to call the `CIMSWinProcess.wsf` script that is on the same computer. If you installed the CIMS Windows Process collector in the default location on the computer, the `CIMSWinProcess.wsf` script is in `C:\Program Files\CIMSLab\Collectors\CIMSWinProcess`. The parameters required for the `CIMSWinProcess.wsf` script are described in *Setting the Parameters for the CIMSWinProcess.wsf Script* on page 3-12.
2. On the CIMS Data Collectors server, set up the collector as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. In the `JobCIMSWinProcess.wsf` script, remove the call to the `CIMSWinProcess.wsf` script.

Setting the Parameters for the CIMSWinProcess.wsf Script

The `CIMSWinProcess.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.
`Feed`	The name of the computer that contains the log file folder. Because this folder must reside on the same server as the `CIMSWinProcess.wsf` script, you can either enter the name of the server or `SELF`. A subfolder with the same name as the server is automatically created in the process definition folder (see the `OutputFolder` parameter). This subfolder is used to store the **CSR files** that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the **CSR files**.
`OutputFolder`	The process definition folder for the collector on the central CIMS Data Collectors server. This is the location of the final **CSR file** that is created by the `Scan.wsf` script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
`LogFolder`	The location of the log file. This folder must be on the same computer as the `CIMSWinProcess.wsf` script.

Running the CIMS Windows Process Collection Process

Run the CIMS Windows Process collection process on the central CIMS Data Collectors server using the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the CIMS Windows Process collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("CIMSWinProcess")
```

Citrix Data Collector

The CIMS Data Collector for Citrix collects data that is contained in the Citrix Resource Manager summary database and creates a log file from the data. This log file provides CPU time and memory used by user, server, and process.

Identifiers and Resources Collected by the Citrix Collector

The Citrix collector creates the following chargeback identifiers and resources from the data collected. These are the identifiers and resources that will appear in the output CSR file, which is processed and loaded into the CIMS Server database.

Identifiers

- UserName (the user that accessed the application or information)
- ServerName (the Citrix server from which the application/information was accessed)
- ProcessName (the process started by the user in the Citrix session)

Resources

- CPU time used
- Memory used

Setting Up the Citrix Collector

CIMS Lab provides a process definition folder named `Citrix`, which contains the following files. These files are described in detail in the following sections.

- The job script for the collector, `JobCitrix.wsf`.
- A conversion definition file for the collector, `JobCitrix.txt`, which defines the identifiers and resources that are collected from the Citrix data.
- A file supplied by Citrix, `SDBReportViews.sql`, which installs views in the Citrix Resource Manager summary database.

The `Citrix` folder must be placed in the `Processes` folder (see *About the Processes Folder* on page 2-8).

To request the Citrix collector, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Editing the JobCitrix.wsf Script

The `JobCitrix.wsf` script controls the collection and processing of the data from the Citrix Resource Manager summary database. Edit line 97 of the script (`strODBCDSNConnection =`) to supply the ODBC connection string for the database.

Defining Resource Rate Codes in the JobCitrix.txt File

The JobCitrix.txt file contains the conversion information required by CIMS Conversion Builder to create a CSR file from the Citrix log file.

Lines 29 and 30 in the JobCitrix.txt file define the resource fields in the log file. Change the rate codes for the resources CPU time and memory to the rate codes that you want to use for these resources (maximum of 8 characters).

For example, change:

```
RSField1=CPUTime RATECODE(MISC)
RSField2=Memory RATECODE(MISC)
```

To:

```
RSField1=CPUTime RATECODE(CTRXCPU)
RSField2=Memory RATECODE(CTRXMEM)
```

You can open the JobCitrix.txt file in a text editor such as Notepad, or you can edit the file in CIMS Conversion Builder (see *Creating a Definition file* on page 12-3).

Adding Resource Rate Codes to the CIMS Rate Table

Because the rate codes for the resources collected by the Citrix collector are defined in the JobCitrix.txt file and not pre-defined by CIMS Lab, you need to add the rate codes to the CIMS Rate table.

Rate codes that do not appear in the CIMS Rate table are not included in CIMS invoices and other reports. You cannot load a CSR file into the CIMS Server database until at least one rate code from the file is added to the CIMS Rate table.

To add rate codes to the CIMS Rate table, start CIMS Server Administrator and click **Chargeback Administration ▸ Chargeback Table Maintenance ▸ Rate Codes**. Follow the instructions in the *CIMS Server Administrator's Guide*.

Installing Views in the Citrix Database

You need to install views supplied by Citrix in the Resource Manager summary database to enable the Citrix collector to extract the required data. To install the views, execute the SDBReportViews.sql file as follows:

1. In Windows, click **Start ▸ Programs ▸ Microsoft SQL Server ▸ Query Analyzer**.
2. In the Connect to SQL Server dialog box, click the server that contains the Resource Manager summary database. In the **Connect using** area, click the authentication method that is used for the database and type a user login name and password as needed.
3. In the SQL Query Analyzer window, click **File ▸ Open.**
4. Browse for the SDBReportViews.sql file, and then click **Open**.
5. Click **Query ▸ Execute**.

Running the Citrix Collector

To run the Citrix collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Citrix collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("Citrix")
```

AS/400 Data Collector

CIMS Lab provides a CIMS Data Collector for AS/400. For instructions on how to configure this collector, contact CIMS Lab (*Chapter 13, Contacting Technical Support*).

4

Database Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for databases. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Microsoft SQL Server 2000 Data Collector

The CIMS Data Collector for Microsoft SQL Server 2000 collects data that is contained in a trace file produced by SQL Server. This trace file provides useful metrics such as:

- SQL Server login name or Windows NT user name.
- Amount of elapsed time taken by an event.
- Amount of CPU time used by an event.
- Number of logical disk reads performed by the server on behalf of the event.
- Number of physical disk writes performed by the server on behalf of the event.

The CIMS Lab stored procedure `CIMSSp_SQLServer2000Trace.sql` is used to create the trace file. Instructions for installing and running this stored procedure are provided in *Enabling SQL Server 2000 Tracing*.

The following sections provide steps for enabling tracing for SQL Server 2000 and for setting up and running the SQL Server collector.

Note • The SQL Server collector supports SQL Server clusters. A cluster refers to a group of two or more servers that work together and represent themselves as a single virtual server to a network.

Creating an Output Folder for Storing the Trace Files

You need to create an output folder for storing trace files from the database server or servers that you want to collect data from before you run the stored procedure `CIMSSp_SQLServer2000Trace.sql`. This folder must be located on the same server as the SQL Server 2000 processing script, `MSSQL2000.wsf`. You should create this folder in a location where you keep data that is backed up.

The location of the trace folder is passed as a parameter from the `JobMSSQL2000.wsf` script to the `MSSQL2000.wsf` script (see *Setting the Parameters for the MSSQL2000.wsf Script* on page 4-10).

Important! • Do not store trace files (*yyyymmdd-hhmmss*.trc) in the `Processes\MSSQL2000\`*feedname* folder. The feed folder should contain only CIMS Server Resource (CSR) files.

Enabling SQL Server 2000 Tracing

The stored procedure `CIMSSp_SQLServer2000Trace` performs the following functions:

- Stops the current SQL trace logging.
- Closes the current trace file in the trace file folder.
- Starts a new trace file in the trace file folder.

Once installed, the stored procedure should be scheduled to run once a day (see *Running the CIMSSp_SQLServer2000Trace Stored Procedure* on page 4-5).

`CIMSSp_SQLServer2000Trace` uses the following built-in Microsoft SQL Server 2000 stored procedures. For more information about these stored procedures, use the link to go to the description on the Microsoft Web site.

- `sp_trace_setstatus`

  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tsqlref/ts_sp_ta-tz_0tnt.asp
- `sp_trace_create`

  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tsqlref/ts_sp_ta-tz_8h49.asp
- `sp_trace_setevent`

  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tsqlref/ts_sp_ta-tz_1c4p.asp

Installing the CIMSSp_SQLServer2000Trace Stored Procedure

To create SQL Server trace files, you need to install the stored procedure `CIMSSp_SQLServer2000Trace` on each server that you want to collect data from.

CIMS Lab provides a script, `InstallSQLTrace.bat`, that you can use to install the stored procedure. (If you installed CIMS Data Collectors in the default location, this script is in `C:\Program Files\CIMSLab\Collectors\MSSQLServer\2000`.) To use `InstallSQLTrace.bat`:

1. In the script file, change the `-d` parameter `CIMSServer` to your database name.
2. Edit the other parameters as needed. For example, change the `-i` parameter if the stored procedure `SQLServer2000Trace.sql` (creates `CIMSSp_SQLServer2000Trace`) is stored in another location.
3. Run the script to place the stored procedure `CIMSSp_SQLServer2000Trace` in the database.

If you have databases on multiple servers, you need to change the parameters as needed and run the script for each server.

Running the CIMSSp_SQLServer2000Trace Stored Procedure

Note • Make sure that you have created a folder for storing trace files before running the stored procedure (see *Creating an Output Folder for Storing the Trace Files* on page 4-3).

There are two methods for running the stored procedure: from the `MSSQL2000.wsf` script (recommended) or using SQL Server Enterprise Manager scheduling tools.

Running the Stored Procedure from the MSSQL2000 Script

Running the stored procedure from the `MSSQL2000.wsf` script uses the schedule that you have set up in Windows Task Scheduler for the SQL Server collector. No additional scheduling is required. The stored procedure on each server is run as part of the collection process and the trace file is "pulled" from the database server into the trace file folder on the collector server.

Note • To run the stored procedure from the `MSSQL2000.wsf` script, each database that you are collecting from must be entered as a data source in the Windows ODBC Data Source Administrator. To create a data source, see *Appendix D, Adding Data Sources*.

To enable the `MSSQL2000.wsf` script to run the stored procedure, you need to provide the database (data source) and database user ID and password (if required) as parameters (see *Setting the Parameters for the MSSQL2000.wsf Script*).

Running the Stored Procedure Using SQL Server Scheduling Tools

If you do no include a data source parameter in the `MSSQL2000.wsf` script, you must run the stored procedure using SQL Server Enterprise Manager scheduling tools. The trace files are then "pushed" from the database server into the trace file folder on the collector server.

Modifying the SQLServer2000Trace Stored Procedure (Optional)

The stored procedure `SQLServer2000Trace.sql` creates the stored procedure `CIMSSp_SQLServer2000Trace` that is installed on the database server(s). (If you installed CIMS Data Collectors in the default location, `SQLServer2000Trace` is in `C:\Program Files\CIMSLab\Collectors\MSSQLServer\2000`.)

`SQLServer2000Trace` defines the event that causes data to be logged to the trace file and the event columns (data) that appear in the trace file (see *SQL Server 2000 Trace File Format*). You can change the event and event columns. However, the columns are defined as identifiers and resources in the `MSSQL2000.wsf` script. If you change the columns, you need to modify and script. You also need to add any new rate codes to the CIMS Rate table as described in the *CIMS Server Administrator's Guide*.

SQL Server 2000 Trace File Format

Data is logged to the trace file when the SQL Server event ID 15, `Logout` (the user logs out of SQL Server), occurs.

The following table describes the event columns that are provided in the trace file.

Column Name	Description/Values
`TextData`	Text value dependent on the event class that is captured in the trace.
`BinaryData`	Binary value dependent on the event class captured in the trace.
`DatabaseID`	ID of the database specified by the `USE <database>` statement, or the default database if no `USE <database>` statement is issued for a given connection. The value for a database can be determined by using the DB_ID function.
`TansactionID`	System-assigned ID of the transaction.
`Reserved`	
`NTUserName`	Microsoft Windows NT user name.
`NTDomainName`	Windows NT domain to which the user belongs.
`ClientHostName`	Name of the client computer that originated the request.
`ClientProcessID`	ID assigned by the client computer to the process in which the client application is running.
`ApplicationName`	Name of the client application that created the connection to an instance of SQL Server. This column is populated with the values passed by the application rather than the displayed name of the program.
`SQLSecurityLoginName`	SQL Server login name of the client.
`SPID`	Server Process ID assigned by SQL Server to the process associated with the client.
`Duration`	Amount of elapsed time (in milliseconds) taken by the event. This data column is not populated by the `Hash Warning` event.
`StartTime`	Time that the event started, when available.

Table 4-1 • SQL Server 2000 Trace File Format

Column Name	Description/Values
EndTime	Time that the event ended. This column is not populated for starting event classes, such as SQL:BatchStarting or SP:Starting. It is also not populated by the Hash Warning event.
Reads	Number of logical disk reads performed by the server on behalf of the event. This column is not populated by the Lock:Released event.
Writes	Number of physical disk writes performed by the server on behalf of the event.
CPU	Amount of CPU time (in milliseconds) used by the event.
ObjectName	Name of object accessed.
DatabaseName	Name of the database specified in the USE <database> statement.
Filename	Logical name of the file name modified.
ObjectOwner	Owner ID of the object referenced.
TargetRoleName	Name of the database or server-wide role targeted by a statement.
TargetUserName	User name of the target of some action.
DatabaseUserName	SQL Server database user name of the client.
ServerName	Name of the instance of SQL Server (either servername or servername\instancename) being traced.

Table 4-1 • SQL Server 2000 Trace File Format (Continued)

Identifiers and Resources Collected from the SQL Server 2000 Trace File

By default, the following fields in the SQL Server 2000 trace file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the SQL Server 2000 processing script, `MSSQL2000.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

Trace File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
—	Feed (passed from the `JobMSSQL2000.wsf` script)	—
BinaryData	EventClass	—
SPID	SPID	—
SQLSecurityLoginName	LoginName	—
ApplicationName	ApplicationName	—
TextData	TextData	—
NTDomainName	NTDomainName	—
NTUserName	User	—
ClientHostName	HostName	—
ClientProcessID	ClientProcessID	—
ServerName	Server	—
DatabaseName	DatabaseName	—
DatabaseID	DatabaseID	—
Resources		
—	MS Windows SQL Server Records This is the number of records in the log file. That is, each time an event ID 15 occurs, a record is added to the log file. This resource is passed from the `MSSQL2000.wsf` script.	SQLREC
Duration	MS Windows SQL Server Duration	SQLDUR
CPU	MS Windows SQL Server CPU	SQLCPU

Table 4-2 • Default SQL Server 2000 Identifiers and Resources

Trace File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Reads	MS Windows SQL Server Reads	SQLREADS
Writes	MS Windows SQL Server Writes	SQLWRITE

Table 4-2 • Default SQL Server 2000 Identifiers and Resources (Continued)

Setting Up the SQL Server 2000 Collector

This section provides information about the scripts specific to the SQL Server collector: `JobMSSQL2000.wsf` and `MSSQL2000.wsf`.

If you installed CIMS Data Collectors in the default location, the `MSSQL2000.wsf` script is in `C:\Program Files\CIMSLab\Collectors\MSSQLServer\2000`.

The `JobMSSQL2000.wsf` script is in `Processes\MSSQL2000` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

For information about other scripts related to the collection process, see *Setting Up the System (System Architecture)* on page 2-3.

Setting Up the JobMSSQL2000.wsf Script

The `JobMSSQL2000.wsf` script calls and passes parameters to the `MSSQL2000.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `MSSQL2000.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the MSSQL2000.wsf Script

The `MSSQL2000.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the trace file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ `yyyymmdd` This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 4-3 • MSSQL2000 Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the trace file folder. Because this folder must reside on the same server as the MSSQL2000.wsf script, you can either enter the name of the server or SELF. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the trace files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
TraceFolder	The location of the .trc file written by the CIMSp_SQLServer2000 stored procedure. This folder must be on the same server as the MSSQL2000.wsf script.
ODBCDSN (optional)	The ODBC DSN for the SQL Server database that you want to trace. Use this parameter if you want the MSSQL2000.wsf script to run the CIMSp_SQLServer2000 stored procedure. If blank, the script assumes that a .trc file already exists in the output folder and the stored procedure is not run. For more information, see *Running the CIMSSp_SQLServer2000Trace Stored Procedure* on page 4-5.
ODBCUserID and ODBCPassword (optional)	The SQL Server user ID and password for the database. These parameters are required only if using the SQL Server user ID to access the database. The default is blank.
RunSp (optional)	This parameter forces the CIMSp_SQLServer2000 stored procedure to run on the server that you want to collect data from.

Table 4-3 • MSSQL2000 Script Parameters (Continued)

Running the SQL Server 2000 Collector

To run the SQL Server 2000 collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the SQL Server collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("MSSQL2000")
```

Microsoft SQL Server 7 Data Collector

CIMS Lab provides a CIMS Data Collector for Microsoft SQL Server 7. For instructions on how to configure this collector, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Oracle Data Collector

The CIMS Data Collector for Oracle (called CIMSWIND) collects data from the event log and from a data file created by the CIMS Oracle Accounting Service. The event log and data file provide useful metrics such as:

- System, user, and database name.
- Amount of CPU time used by an Oracle session.
- Memory used in the User Global Area and Program Global Area.
- Number of commits performed by the user.
- Number of reads from and writes to the database files.

The following sections provide steps for setting up and running the Oracle collector and enabling Oracle logging using the CIMS Oracle Accounting Service.

Setting Up the CIMSWIND Collector

Note • This section provides steps for setting up the CIMSWIND collector for Oracle *and* DB2 data collection.

If you are running CIMSWIND in a client/server environment, you need to install CIMSWIND on the client(s) and the server. The following post-installation instructions are applicable to both client and server unless noted otherwise. These instructions assume that CIMSWIND is installed in the default location `C:\Program Files\CIMSLab\Collectors`.

1 Verify that the following system environment variables have been established:

```
ARSAP_DATA=C:\PROGRA~1\CIMSLab\Collectors\CIMSWIND\DATA

ARSAP_HELP=C:\PROGRA~1\CIMSLab\Collectors\CIMSWIND\HELP

ARSAP_HOME=C:\PROGRA~1\CIMSLab\Collectors\CIMSWIND

ARSAP_LOG=C:\PROGRA~1\CIMSLab\Collectors\CIMSWIND\LOG
```

For Windows NT Server, click **Control Panel ▸ System ▸ Environment tab**.

For Windows 2000 Server, click **Control Panel ▸ System ▸ Advanced tab ▸ Environment Variables**.

2 In the `CIMSWIND\Data` folder, rename the file `Sample_NT_config_par.bat` to `NT_config_par.bat`. You need to modify this configuration file and renaming the file prevents it from being overwritten when you upgrade to a new version of CIMS Data Collectors.

3. Load the licensing information from the license PAK provided by CIMS Lab as follows:

   a. At the command prompt, run `CIMSWIND\Etc\NT_add_license.bat`.

   b. When prompted, enter the values contained in the license PAK exactly as provided by CIMS Lab. The values are case-sensitive.

   If you do not have your license PAK, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

4. Verify that the security options for the following audit polices have been set to Success/Failure:

   - For Windows NT Server, click **Start ▸ Programs ▸ Administrative Tools ▸ User Manager Policies ▸ Audit**.

     ```
     Logon and Logoff

     Restart, Shutdown, and System

     Process Tracking
     ```

   - For Windows 2000 Server, click **Start ▸ Programs ▸ Administrative Tools ▸ Local Security Policy ▸ Local Policies ▸ Audit Policy**.

     ```
     Audit logon events

     Audit process tracking

     Audit system events
     ```

5. In the Windows Event Viewer, verify that the maximum log size for all event logs is set to a size sufficient to hold event records for more than one day. This size may vary depending on the usage on any particular platform. The default setting of 512 KB is usually sufficient.

6. In Windows Scheduled Tasks, schedule the following scripts:

   - `CIMSWIND\Etc\NT_nightly.bat`. This nightly collection script should be scheduled to run nightly around 1 a.m. This script calls `NT_arsap_nightly.bat`, which produces the CIMSWIND Accounting File. The CIMSWIND Accounting File contains the combined data collected from the event log and database data file.

     Note that in a client/server environment, this script is not required on the CIMSWIND server unless you are collecting data from the server.

   - `CIMSWIND\Etc\NT_process.bat`. This nightly consolidation script should be scheduled only on the CIMSWIND server and not on clients. This script calls `NT_process_nightly.bat`, which consolidates the collected CIMSWIND Accounting Files and produces CSR files. This script should be scheduled to run nightly around 5 a.m.

7 On the CIMSWIND server, do the following:

- In `CIMSWIND\Accounting`, create a folder for each client computer. If this is a stand-alone implementation, create a folder for this server. This folder is used to store the CIMSWIND Accounting Files (filename `acc_yyyymmdd.dat`).

  The folder must have the same name as the client or server name. For example, if you are creating a folder for a client computer named `ClientB`, the folder name must be `ClientB`.

- Open `CIMSWIND\Data\A_node.par` and add name of each client. If this is stand-alone implementation, add the name of this server. Enter the names on separate lines.

8 Set the following environment variable values in `NT_config_par.bat`:

- `set ARSAP_SERVER=<server name>`

  If you are setting this value on a client, this is the name of the CIMSWIND server. If you are setting this value on a stand-alone server, this is the name of the server.

- `set DEST=<destination path>`

  This variable sets the path for the destination folder for the CIMSWIND Accounting Files on the CIMSWIND server. These are the folders that you created in Step 7.

  If you are using CIMSWIND in a stand-alone environment, the CIMSWIND Accounting Files are stored in a folder with the same name as the server. For example, if the server name is `ServerA`, the environment variable would be `DEST=C:\PROGRA~1\CIMSLAB\Collectors\CIMSWIND\Accounting\ServerA`.

  If you are using CIMSWIND in a client/server environment, the CIMSWIND Accounting Files are stored in a folder with the same name as the client. On the client, a UNC is recommended for the folder path. For example, if the client name is `ClientB`, the environment variable might be `DEST=\\SERVERA\CIMSWIND_ACCOUNTING\CLIENTB` where the share `CIMSWIND_ACCOUNTING` was created for the `C:\PROGRA~1\CIMSLAB\Collectors\CIMSWIND\Accounting` on `ServerA`.

The `ARSAP_SERVER` and `DEST` settings are commented by default. Make sure that you remove the comment.

Creating a Process Definition Folder for Oracle Data Collection

You need to create a process definition folder and script for the Oracle collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `WinOracle`, the name of the job script should be `JobWinOracle.wsf`.

For more information about the job script, see *Job Script* on page 2-9.

Enabling Oracle Logging

To enable logging for Oracle, you need to use the CIMS Oracle Accounting Service. The following are instructions for starting this service for one Oracle instance. If you have multiple Oracle instances on your computer, you need to repeat these steps for each instance.

The following instructions assume that CIMSWIND is installed in the default installation location `C:\Program Files\CIMSLab\Collectors` and that accounting for an Oracle instance named `ORCL` is being enabled.

1. Copy the CIMS Oracle Accounting Service executable `CIMSWIND\BIN\NT_dbao.exe` to create a new executable named `NT_dbao_ORCL.exe` (includes the name of the Oracle instance to be tracked).

2. Install the CIMS Oracle Accounting Service in Windows Services. At the command prompt, go to `C:\Program Files\CIMSLab\Collectors\CIMSWIND\Bin` and execute the command:

   ```
   NT_dbao_ORCL -install
   ```

   Note that the service can be removed from Services with the command:

   ```
   NT_dbao_ORCL -remove
   ```

3. Create an Oracle user account to be used by the CIMS Oracle Accounting Service for connecting to the Oracle instance. In the following example, the user name is `cims` and the password is `acct123`:

   ```
   SQL> CREATE USER cims IDENTIFIED BY acct123;
   ```

4 The Oracle user `cims` must be able to select the following ORACLE instance `V$` system tables:

```
V$DATABASE

V$PROCESS

V$SESSION

V$SESSTAT

V$STATNAME
```

An SQL script, `CIMSWIND\Etc\Oracle\arsap_view.sql`, is included in the installation. This script creates an Oracle role called `ARSAP_VIEW` with the necessary privileges. The script grants the role to the Oracle user `cims`. The Oracle DBA can run this script after the Oracle user has been created.

5 Create a CIMS DB Instance Record for this Oracle instance. To create this record, run the CIMSWIND setup utility, `CIMSWIND\Bin\NT_setup,exe`.

At the `SETUP>` prompt, enter the command:

```
SETUP>add/dbinst/dbtype=ORACLE/user=cims/password=acct123/frequency=60 ORCL
```

Where `60` indicates a sample frequency of every 60 seconds.

6 Start the CIMS ORACLE Accounting Service either from Services or from the following command at the command prompt:

```
NET START "CIMS/NT Oracle DB Collector-ORCL"
```

7 In the `CIMSWIND\Data\NT_config_par.bat` script, set the following environment variables:

- `set A_ORACLE_ACCT=Y`

  Setting this variable to `Y` instructs the `NT_arsap_nightly.bat` script to include the CIMSWIND Oracle Accounting File in the files collected nightly.

- `set GEN_ORACLE=Y`

  Setting this variable to `Y` instructs the `NT_process_nightly.bat` script to generate Oracle CSR files for input into CIMS Server.

- `CS_GCS_DEST=<destination folder for CSR files>`

  This is the destination folder for the generated CSR files. You need to change this location to the job process folder that you created in *Creating a Process Definition Folder for Oracle Data Collection* on page 4-16.

Note • `NT_config_par.bat` **is shipped as** `Sample_NT_config_par.bat`**. You should have renamed the script in** Step 2 **on page 4-13**.

8 Schedule the script `CIMSWIND\Etc\CIMS_start_db_svc.bat` to be run when the computer is started. This script automatically starts the CIMS Oracle Accounting Service.

This script requires modification as indicated in the comments at the beginning of the script. Rename this script so that the modification is not overwritten when you upgrade to a new version of CIMS Data Collectors.

Resources Collected

By default, the following resources in the event log and Oracle data file are defined as the chargeback resources in CIMS Server.

Resource	Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Event Log Resources		
Logins to the system	MS Windows Logins	LLT101
Connect time on the system in hours	MS Windows Connect Time (hours)	LLT102
Number of images executed	MS Windows Image Count	LLT103
Time spent executing	MS Windows Image Time (hours)	LLT104
Oracle Resources		
Number of Oracle sessions	MS Windows Oracle Logins	LLW101
CPU utilized in Oracle sessions	MS Windows Oracle Session CPU (minutes)	LLW102
Amount of time a user is connected to Oracle	MS Windows Oracle Connect (hours)	LLW103
Memory used in the User Global Area	MS Windows Oracle UGA Memory	LLW104
Memory used in the Program Global Area	MS Windows Oracle PGA Memory	LLW105
Oracle Recursive CPU – CPU used updating internal tables	MS Windows Oracle Rec CPU (minutes)	LLW106
Commits performed by the user	MS Windows Oracle User Commits	LLW107

Table 4-4 • Default Oracle Resources

Resource	Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Reads from database files	MS Windows Oracle Physical Reads	LLW108
Writes to database files	MS Windows Oracle Physical Writes	LLW109
Write requests to database files	MS Windows Oracle Write Requests	LLW110
Memory utilized to perform an external sort	MS Windows Oracle Disk Sorts	LLW111
Messages sent to perform database updates	MS Windows Oracle Messages Sent	LLW112
Messages received to update database	MS Windows Oracle Messages Received	LLW113

Table 4-4 • Default Oracle Resources (Continued)

Running the Oracle Collector

To run the Oracle collector, you need to run the following scripts:

- `CIMSWIND\Etc\NT_nightly.bat`. This script should be scheduled to run nightly around 1 a.m. Note that in a client/server environment, this script is not required on the CIMSWIND server unless you are collecting data from the server.
- `CIMSWIND\Etc\NT_process.bat`. This script should be scheduled to run nightly around 5 a.m. The script should be scheduled only on the CIMSWIND server and not on clients.
- `Nightly.bat` (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). This script should be scheduled to run nightly after `NT_process.bat` has run. For instructions for running `Nightly.bat`, see *Appendix C, Running Batch Scripts*.

Make sure that the CIMSWIND collector is set up correctly as described on page 4-13. The name of the process definition folder for the Oracle collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("WinOracle")
```

Where `WinOracle` is an example process definition folder name.

DB2 Data Collector

The CIMS Data Collector for DB2 (called CIMSWIND) collects data from the event log and from a data file created by the CIMS DB2 Accounting Service. The event log and data file provide useful metrics such as:

- System, user, and database name.
- System and user CPU utilization.
- Number of read and write operations that do not use a buffer pool.
- Buffered pool data writes and logical and physical reads.
- Buffered pool index writes and logical and physical reads.
- Number of row delete, insert, and update operations.

The following sections provides instructions for creating a DB2 process definition folder, running the DB2 collector, and enabling DB2 logging.

You also need to complete the instructions for setting up the CIMSWIND collector that are provided in the *Oracle Data Collector* section. These instructions are applicable to both Oracle and DB2.

Setting up the CIMSWIND Collector

See *Setting Up the CIMSWIND Collector* on page 4-13.

Creating a Process Definition Folder for DB2 Data Collection

You need to create a process definition folder and script for the DB2 collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `WinDB2`, the name of the job script should be `JobWinDB2.wsf`.

For more information about the job script, see *Job Script* on page 2-9.

Enabling DB2 Logging

To enable logging for DB2, you need to use the CIMS DB2 Accounting Service. The following are instructions for starting this service for one DB2 instance. If you have multiple DB2 instances on your computer, you need to repeat these steps for each instance.

The following instructions assume that CIMSWIND is installed in the default installation location `C:\Program Files\CIMSLab\Collectors\CIMSWIND` and that accounting for a DB2 instance named `DB2MPP` is being enabled.

1. Copy the CIMS DB2 Accounting Service executable `CIMSWIND\BIN\NT_dbadb2.exe` to create a new executable named `NT_dbadb2_DB2MPP.exe` (includes the name of the DB2 instance to be tracked).

2. Install the CIMS DB2 Accounting Service in Windows Services. At the command prompt, go to `C:\Program Files\CIMSLab\Collectors\CIMSWIND\Bin` and execute the command:

   ```
   NT_dbadb2_DB2MPP -install
   ```

   Note that the service can be removed from Services with the command:

   ```
   NT_dbadb2_DB2MPP -remove
   ```

3. Set the following DB2 Monitor switches to on for the instance:

   - `BUFFERPOOL`
   - `LOCK`
   - `SORT`
   - `UOW`

4. Create a CIMS DB Instance Record for this DB2 instance. To create this record, run the CIMSWIND setup utility, `CIMSWIND\Bin\NT_setup.exe`.

   At the `SETUP>` prompt, enter the command as shown in the following example where a user named `cims` has sufficient privileges to access DB2 monitoring information. The password for this user is `acct123` and the sample frequency is every 60 seconds.

   ```
   SETUP>add/dbinst/dbtype=DB2/user=cims/password=acct123/frequency=60 DB2MPP
   ```

5. Start the CIMS DB2 Accounting Service either from Services or from the following command at the command prompt:

   ```
   NET START "CIMS/NT DB2 Collector-DBMPP"
   ```

6 In the `CIMSWIND\Data\NT_config_par.bat` script, set the following environment variables:

- `set A_DB2_ACCT=Y`

  Setting this variable to `Y` instructs the `NT_arsap_nightly.bat` script to include the CIMSWIND DB2 Accounting File in the files collected nightly.

- `set GEN_DB2=Y`

  Setting this variable to `Y` instructs the `NT_process_nightly.bat` script to generate DB2 CSR files for input into CIMS Server.

- `CS_GCS_DEST=<destination folder for CSR files>`

  This is the destination folder for the generated CSR files. You need to change this location to the job process folder that you created in *Creating a Process Definition Folder for DB2 Data Collection* on page 4-20.

Note • `NT_config_par.bat` **is shipped as** `Sample_NT_config_par.bat`**. You should have renamed the script in** Step 2 **on page 4-13****.**

7 Schedule the script `CIMSWIND\Etc\CIMS_start_db_svc.bat` to be run when the computer is started. This script automatically starts the CIMS DB2 Accounting Service.

This script requires modification as indicated in the comments at the beginning of the script. Rename this script so that the modification is not overwritten when you upgrade to a new version of CIMS Data Collectors.

Resources Collected

By default, the following resources in the event log and DB2 data file are defined as the chargeback resources in CIMS Server.

Resource	Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Event Log Resources		
Logins to the system	MS Windows Logins	LLT101
Connect time on the system in hours	MS Windows Connect Time (hours)	LLT102
Number of images executed	MS Windows Image Count	LLT103
Time spent executing	MS Windows Image Time (hours)	LLT104
DB2 Resources		
SQL commit statements that have been attempted	MS Windows DB/2 Commit SQL STMTS	LLX101
Number of deadlocks that have occurred	MS Windows DB/2 Deadlocks	LLX102
The number of read operations that do not use the buffer pool	MS Windows DB/2 Direct Reads	LLX103
The number of write operations that do not use the buffer pool	MS Windows DB/2 Direct Writes	LLX104
Rollbacks initiated by the database manager due to a deadlock	MS Windows DB/2 Int Deadlock Rollbacks	LLX105
Elapsed time waiting for a lock	MS Windows DB/2 Lock Wait Time	LLX106
The number of times a user connects to the database	MS Windows DB/2 Logins	LLX107
Buffered pool data logical reads	MS Windows DB/2 PD Lreads	LLX108
Buffered pool data physical reads	MS Windows DB/2 PD Preads	LLX109

Table 4-5 • Default DB2 Resources

Resource	Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Buffered pool data writes	MS Windows DB/2 PD Writes	LLX110
Buffered pool index logical reads	MS Windows DB/2 PI Lreads	LLX111
Buffered pool index physical reads	MS Windows DB/2 PI Preads	LLX112
Buffered pool index writes	MS Windows DB/2 PI Writes	LLX113
SQL rollback statements attempted	MS Windows DB/2 Rollback SQL Statements	LLX114
The number of row deletion operations	MS Windows DB/2 Rows Deleted	LLX115
The number of row inserted operations	MS Windows DB/2 Rows Inserted	LLX116
The number of row select/returned to the application	MS Windows DB/2 Rows Selected	LLX117
The number of row updated operations	MS Windows DB/2 Rows Updated	LLX118
System CPU used by the database manager process	MS Windows DB/2 SCPU (minutes)	LLX119
Number of sorts that ran out of sort heap	MS Windows DB/2 Sort Overflows	LLX120
Number of sorts executed	MS Windows DB/2 Total Sorts	LLX121
LLX122	MS Windows DB/2 UCPU (minutes)	User CPU used by the database manager process
LLX123	MS Windows DB/2 UOW Log Space Used	The amount of log space (in bytes) used in the current unit

Table 4-5 • Default DB2 Resources (Continued)

Running the DB2 Collector

To run the DB2 collector, you need to run the following scripts:

- `CIMSWIND\Etc\NT_nightly.bat`. This script should be scheduled to run nightly around 1 a.m. Note that in a client/server environment, this script is not required on the CIMSWIND server unless you are collecting data from the server.
- `CIMSWIND\Etc\NT_process.bat`. This script should be scheduled to run nightly around 5 a.m. The script should be scheduled only on the CIMSWIND server and not on clients.
- `Nightly.bat` (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). This script should be scheduled to run nightly after `NT_process.bat` has run. For instructions for running `Nightly.bat`, see *Appendix C, Running Batch Scripts*.

Make sure that the CIMSWIND collector is set up correctly as described on page 4-13. The name of the process definition folder for the DB2 collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("WinDB2")
```

Where `WinDB2` is an example process definition folder name.

Sybase Data Collector

CIMS Lab provides a CIMS Data Collector for Sybase. For instructions on how to configure this collector, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Database Size Data Collector (DBSpace)

The CIMS Data Collector for database size, DBSpace, collects data regarding the size of Microsoft SQL Server and Sybase databases. The DBSpace collector uses the stored procedure `sp_helpdb`. To run this collector, you need authority to run `sp_helpdb`.

The following sections provide instructions for setting up and running the DBSpace collector.

Identifiers and Resources Collected by the DBSpace Collector

By default, the following data collected by the DBSpace collector is defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the DBSpace processing script, `DBSpace.wsf`. The rate codes assigned to the database size resource *are not* pre-loaded in the CIMS Rate table and must be added to the table as described in the *CIMS Server Administrator's Guide*.

Identifiers

- Feed (this is passed from the job script for the collector)
- Database
- Owner
- DBID

Resource Rate Codes

- MSDBSIZE (SQL Server database size in megabytes)
- SYDBSIZE (Sybase database size in megabytes)

Setting Up the DBSpace Collector

This section provides information about creating the process definition folder and job script for the DBSpace collector.

This section also provides information about the DBSpace.wsf script. If you installed CIMS Data Collectors in the default location, the DBSpace.wsf script is in C:\Program Files\CIMSLab\Collectors\DBSpace.

For information about other scripts related to the collection process, see the *Scripts Architecture* section beginning on page 2-13

Creating a Process Definition Folder and Job Script for the DBSpace Collector

You need to create a process definition folder and script for the DBSpace collector in the Processes folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with Job followed by the name of the process definition folder. For example, if the folder name is DBSpace, the name of the job script should be JobDBSpace.wsf.

The job script for DBSpace calls and passes parameters to the DBSpace.wsf, Scan.wsf, ProcCIMS.wsf, and CleanUp.wsf scripts. The parameters required for the DBSpace.wsf script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13 .

Setting the Parameters for the DBSpace.wsf Script

The DBSpace.wsf script requires the parameters shown in the following table.

Parameter	Description/Values
LogDate	This parameter specifies the date that will appear in the CSR file. The values are: ■ preday (previous day) ■ premon (previous month) ■ rndate (current day) ■ curday (current day and previous day) ■ curmon (current month) ■ yyyymmdd This parameter is entered in the command line when running the collector.
RetentionFlag	This parameter is for future use.

Table 4-6 • DBSpace Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the database that you want to collect size data from or SELF if the database is on the same server as the DBSpace.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created by the collector (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector on the central CIMS Data Collectors server. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
DBType	The database type. The values are: ■ MS (SQL Server) ■ SY (Sybase)
ODBCDSN	The ODBC DSN for the database that you want to collect from. Note that each database that you are collecting from must be entered as a data source in the Windows ODBC Data Source Administrator. To create a data source, see *Appendix D, Adding Data Sources*.
ODBCUserID and ODBCPassword (optional)	The user ID and password for the database. These parameters are required only if using the database user ID to access the database. The default is blank.

Table 4-6 • DBSpace Script Parameters

Running the DBSpace Collector

To run the DBSpace collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the DBSpace collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("DBSpace")
```

Where `DBSpace` is an example process definition folder name.

5

E-mail Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for e-mail applications. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Microsoft Exchange Server 5.5 and 2000 Data Collector

The CIMS Data Collector for Microsoft Exchange Server 5.5 and 2000 collects and processes data that is contained in a log file produced by Exchange Server. This log file provides useful metrics such as the number of e-mail messages and bytes sent and received by user.

The following sections provide instructions for enabling logging for Exchange Server 5.5 or 2000 and for setting up and running the Exchange Server collector.

Enabling Exchange Server 5.5 Logging

The following provides an example of enabling message tracking logging for Exchange Server 5.5 components on Windows NT Server 4.0. Refer to the Microsoft documentation for instructions on how to enable logging on other platforms.

Enabling Message Tracking on MTAs

To enable message tracking on all Message Transfer Agents (MTAs) on a site:

1. In the Microsoft Exchange Administrator window, click **Configuration** or **Information Site Configuration**.
2. Double-click **MTA Site Configuration**.

   The MTA Site Configuration Properties dialog box appears.
3. On the **General** tab, select the **Enable message tracking** check box.
4. Restart the MTAs or restart the computer.

Enabling Message Tracking on a Microsoft Mail Connector

You must enable message tracking separately on each mail connector on a site. To enable message tracking:

1. In the Microsoft Exchange Administrator window, click **Connections**.
2. Double-click a mail connector.

   The connector properties dialog box appears.
3. On the **Interchange** tab, select the **Enable message tracking** check box.
4. Restart the mail connector or restart the computer.

Enabling Message Tracking on the Internet Mail Service

You must enable message tracking separately on each Internet Mail Service on a site. To enable message tracking:

1. In the Microsoft Exchange Administrator window, navigate to and click **Connections**.
2. Double-click a Internet Mail Service.

   The Internet Mail Service Properties dialog box appears.
3. On the **Internet Mail** tab, select the **Enable message tracking** check box.
4. Restart the Internet Mail Service or restart the computer.

For more information about Exchange Server 5.5 logging, refer to the Microsoft documentation.

Exchange Server 5.5 Log File Name and Format

The Exchange Server 5.5 message tracking log is stored in `exchsrvr\tracking.log`. Each day, a new log is created that records one day's activities on the server. Each daily log is named by the date on which it was created in *`yyyymmdd`*`.log` format.

The following table describes the record fields in the Exchange Server 5.5 log file.

Field Name	Description/Values
Message ID or MTS-ID	Message ID is a unique identifier assigned to the message by Exchange Server. It stays with the message from its origination to delivery or transfer from the network. Messages from foreign systems include a message transfer system-ID (MTS-ID) that uniquely identifies the component that transported the message.
Event #	A number that represents the event type.
Date/Time	Date and time of the event.
Gateway Name	Name of the gateway or connector that generated the event. If no gateway was involved, the field is blank.
Partner Name	Name of the messaging service associated with the event. In Exchange Server, the partner name is the MTA or Information Store.
Remote ID	Message ID used by the gateway.
Originator	Distinguished name of the originating mailbox, if known.

Table 5-1 • Exchange Server 5.5 Log File Format

Field Name	Description/Values
Priority	Priority set by the sender. 0 = Normal 1= High -1 = Low
Length	Message length in bytes.
Seconds	Transport time in seconds. Not used by Exchange Server. The value in this field is 0 or blank.
Cost	Cost per second for message transfer. Not used by Microsoft Exchange Server. The value in this field is always 1.
Subject-ID or Report MTS-ID	This field is blank (empty) for normal messages. For reports its value is the Report MTS-ID.
Recipients	Number of recipients.
Recipient Name	Distinguished name of the recipient of the message or a proxy address. This field is separated from the previous field by a line feed and is repeated for each recipient. Because this field is separated by a line, the Exchange Server collector recognizes this field as a record.
Recipient Report Status	A number representing the result of an attempt to deliver a report to the recipient. Delivered = 0 Not delivered = 1 This is used only for reports. On other events, it is blank. This field is repeated for each recipient.

Table 5-1 • Exchange Server 5.5 Log File Format (Continued)

Identifiers and Resources Collected from the Exchange Server 5.5 Log File

By default, the Exchange Server 5.5 collector gathers data from records that contain the following event types in the Event # field:

- 7—Message transfer out
- 9—Message delivered

Depending on the event type, the following fields in the Exchange Server log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the Exchange Server 5.5 processing script, `MSExchange55.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

If you want to gather data for other event types, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Event #	Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers			
7 or 9	—	Feed (passed from the `JobMSExchange.wsf` script)	—
7	Originator	User	—
9	Recipient Name **Note:** Only local recipients are collected.	User	—
Resources			
7	— A value of 1 is automatically assigned.	MS Exchange Emails Sent	EXEMSNT
7	Length	MS Exchange Bytes Sent	EXBYSNT
9	— A value of 1 is automatically assigned for each local recipient record.	MS Exchange Emails Received	EXEMRCV
9	Length	MS Exchange Bytes Received	EXBYRCV

Table 5-2 • Default Exchange Server 5.5 Identifiers and Resources

Enabling Exchange 2000 Server Logging

The following provides an example of enabling message tracking logging for Exchange 2000 Server.

1. In the Exchange System Manager window, double-click **Server**.
2. Right-click a server, and then click **Properties**.
3. On the **General** tab, click the **Enable message tracking** check box.

For more information about Exchange 2000 Server logging, refer to the Microsoft documentation.

Exchange 2000 Server Log File Name and Format

The Exchange 2000 Server message tracking log is stored in `Exchsrvr\servername.log` in which `servername` is the name of your Exchange server. Each day, a new log is created that records one day's activities on the server. Each daily log is named by the date on which it was created in `yyyymmdd.log` format.

The following table describes the record fields in the Exchange 2000 Server log file.

Field Name	Description/Values
Date	Date of the event.
Time	Time of the event.
Client IP	IP address of the sending client or system.
Client-Hostname	Host name of the sending client system.
Partner-Name	Name of the messaging service associated with the event. In Exchange Server, the partner-name is the MTA or Information Store.
Server-Hostname	Host name of the server making the log entry.
Server IP	IP address of the server making the log entry.
Recipient Address	Name of message recipient or a proxy address. This field is separated from the previous field by a line feed and is repeated for each recipient. Because this field is separated by a line, the Exchange Server collector recognizes this field as a record.
Event ID	A number that represents the event type.
MSGID	Message ID is a unique identifier assigned to the message by Exchange Server. It stays with the message from its origination to delivery or transfer from the network.

Table 5-3 • Exchange Server 2000 Log File Format

Field Name	Description/Values
Priority	Priority set by the sender. 0 = Normal 1= High -1 = Low
Recipient Report Status	The number of attempts required to deliver a report to the recipient, in which Delivered = 0 and Not delivered = 1. This field is separated from the previous field by a line feed and is repeated for each recipient. Because this field is separated by a line, the Exchange Server collector recognizes this field as a record.
Total-Bytes	Message length in bytes.
Number-Recipients	Number of recipients.
Origination-Time	Time in seconds it took to deliver the message.
Encryption	The encryption type of the message body. 0 = No encryption 1= Message is signed 2 = Message is encrypted Encryption is tracked for each message, not for each recipient.
Service Version	Version of the service making the log entry.
Linked MSGID	If there is a message ID (MSGID) from another service, it is provided to link the message across services.
Message Subject	The subject message, truncated to 106 bytes.
Sender Address	Primary address of the originating mailbox, if known. The address can be an SMTP address, X.400 address, or a domain name, depending on the transport.

Table 5-3 • Exchange Server 2000 Log File Format (Continued)

Identifiers and Resources Collected from the Exchange 2000 Server Log File

By default, the Exchange 2000 Server collector gathers data from records that contain the following event types in the Event ID field:

- 7—Message transfer out
- 9—Message delivered

Depending on the event type, the following fields in the Exchange 2000 Server log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the Exchange 2000 Server processing script, `MSExchange2000.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

If you want to gather data for other event types, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

Event #	Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers			
7 or 9	—	Feed (passed from the `JobMSExchange.wsf` script)	—
7	Sender Address	User	—
9	Recipient Address **Note:** Only local recipients are collected.	User	—
Resources			
7	— A value of 1 is automatically assigned.	MS Exchange Emails Sent	EXEMSNT
7	Total-Bytes	MS Exchange Bytes Sent	EXBYSNT
9	— A value of 1 is automatically assigned for each local recipient record.	MS Exchange Emails Received	EXEMRCV
9	Total-Bytes	MS Exchange Bytes Received	EXBYRCV

Table 5-4 • Default Exchange 2000 Server Identifiers and Resources

Setting Up the Exchange Server Collector

This section provides information about the scripts specific to the Exchange Server collector: `JobMSExchange.wsf`, `MSExchange55.wsf`, and `MSExchange2000.wsf`.

If you installed CIMS Data Collectors in the default location, `MSExchange55.wsf` is in `C:\Program Files\CIMSLab\Collectors\MSExchange\55` and `MSExchange2000.wsf` is in `MSExchange\2000`.

The `JobMSExchange.wsf` script is in `Processes\MSExchange` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

For information about other scripts related to the collection process, see the *Scripts Architecture* section beginning on page 2-13.

Setting Up the JobMSExchange.wsf Script

The `JobMSExchange.wsf` script calls and passes parameters to the `MSExchange.wsf` (5.5 or 2000), `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `MSExchange.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Note • By default, `JobMSExchange.wsf` calls the `MSExchange55.wsf` processing script. If you want to call the `MSExchange2000.wsf` script, you need to modify the job script.

Setting the Parameters for the MSExchange.wsf Script

The `MSExchange.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 5-5 • MSExchange Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the MSExchange.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CIMS Server Resource (CSR) files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 5-5 • MSExchange Script Parameters (Continued)

Running the Exchange Server Collector

To run the Exchange Server collector, use the Nightly.bat program (this is shipped as SampleNightly.bat, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Exchange Server collector must be included in the list of jobs in the Nightly.wsf script (this is shipped as SampleNightly.wsf, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("MSExchange")
```

Microsoft Outlook Web Access Data Collection

Outlook Web Access enables you to read and send e-mail messages via a Web interface directly to Exchange Server. Therefore, the metrics provided by the standard IIS logs provide information about Web access of the Outlook Web Access page. The metrics provided by the standard Exchange Server logs capture the e-mail messages sent and received.

For information about Exchange Server log files, see *Microsoft Exchange Server 5.5 and 2000 Data Collector* on page 5-2. For information about IIS log files, see *Microsoft Internet Information Services (IIS) Data Collector* on page 6-3.

Lotus Notes Data Collector

CIMS Lab provides a CIMS Data Collector for Lotus Notes. For instructions on how to configure this collector, contact CIMS Lab (*Chapter 13, Contacting Technical Support*).

6

Internet Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for Internet applications. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Microsoft Internet Information Services (IIS) Data Collector

The CIMS Data Collector for Microsoft IIS collects data that is contained in a log file produced by IIS. This log file provides useful metrics such as:

- Bytes sent from a client IP address to a server IP address.
- Bytes sent from a server IP address to a client IP address.
- User name and IP address, IIS site name, and server name and IP address.

You can process log files for IIS Web and FTP sites and the SMTP server.

The following sections provide instructions for enabling logging for IIS and for setting up and running the IIS collector.

Enabling IIS Logging

The following are instructions for enabling logging for IIS 5.1 on the Windows 2000 Server operating system. Refer to the Microsoft documentation for instructions on how to enable logging on other platforms. You need to enable logging for each Web and FTP site and SMTP server individually.

1. In the Windows Internet Information Services window, right-click the site or server, and then click **Properties**.

   The properties dialog box appears.

2. On the **Web Site** tab, select **Enable Logging** and click **W3C Extended Log File Format** from the **Active log format** list.

3. Click **Apply**, and then click **Properties**.

   The Extended Logging Properties dialog box appears.

4. On the **General Properties** tab, set the general properties such as the log schedule (daily, weekly, monthly, etc.) and log file location.

5. On the **Extended Properties** tab, select the properties that you want to log. CIMS Lab recommends that you select all **Extended Properties**. You can also select **Process Accounting** properties; however, this information is not useful for chargeback and is not written to the CIMS Server Resource (CSR) file.

6. Click **OK** to save the settings and close the dialog box, and then click **OK** again to close the properties dialog box.

For more information about IIS logging, refer to the Microsoft documentation.

IIS Log File Format

The following table describes the record fields in the IIS log file.

Field Name	Description/Values
`date`	The date that the action occurred.
`time`	The time that the action occurred.
`c-ip` (client IP address)	The IP address of the client that accessed the server.
`cs-username` (user name)	The name of the authenticated user who accessed the server. This does not include anonymous users, which are represented by a hyphen (`-`).
`s-sitename` (service name)	The Internet service and instance number that was accessed by the client.
`s-computername` (server name)	The name of the server on which the log entry was generated.
`s-ip` (server IP address)	The IP address of the server on which the log entry was generated.
`s-port` (server port)	The port number the client was connected to.
`cs-method` (method)	The action the client was trying to perform (for example, a GET method).
`cs-uri-stem` (URI stem)	The resource accessed (for example, `Default.htm`).
`cs-uri-query` (URI query)	The query, if any, the client was trying to perform.
`sc-status` (protocol status)	The status of the action, in HTTP or FTP terms.
`sc-win32-status` (protocol status)	The status of the action, in terms used by Windows.
`sc-bytes` (bytes sent)	The number of bytes sent by the server.
`cs-bytes` (bytes received)	The number of bytes received by the server.
`time-taken`	The length of time the action took.
`cs-version` (protocol version)	The protocol (HTTP, FTP) version used by the client. For HTTP, this will be either HTTP 1.0 or HTTP 1.1.
`cs-host` (host)	Displays the content of the host header.

Table 6-1 • IIS Log File Format

Field Name	Description/Values
`cs(User-Agent)` (user agent)	The browser used on the client.
`cs(Cookie)` (cookie)	The content of the cookie sent or received, if any.
`cs(Referer)`	The previous site visited by the user. This site provided a link to the current site.

Table 6-1 • IIS Log File Format (Continued)

Identifiers and Resources Collected from the IIS Log File

By default, the following fields in the IIS log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the in the IIS processing script, `MSIIS.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
—	Feed (passed from the `JobMSIIS.wsf` script)	—
	c_ip	—
	c_ip0	—
`c-ip`	c_ip1	—
	c_ip2	—
	c_ip3	—
`cs-username`	User	—
`s-sitename`	sitename	—
`s-computername`	Server	—
	s_ip	—
	s_ip0	—
`s-ip`	s_ip1	—
	s_ip2	—
	s_ip3	—
`s-port`	s–port	—
`cs-uri-stem`	TOPDIR	—
`cs-uri-stem`	NEXTDIR	—
`cs-host`	cs–host	—

Table 6-2 • Default IIS Identifiers and Resources

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
FTP Resources		
cs-bytes	IIS FTP Bytes Received	FCSBytes
sc-bytes	IIS FTP Bytes Sent	FSCBytes
sc-status	IIS FTP Successful Protocol Status 2xx	FIIS-2
sc-status	IIS FTP Redirection Protocol Status 3xx	FIIS-3
sc-status	IIS FTP Client Error Protocol Status 4xx	FIIS-4
sc-status	IIS FTP Server Error Protocol Status 5xx	FIIS-5
time-taken	IIS FTP Time Taken	FTimeTkn
SMTP Resources		
cs-bytes	IIS SMTP Bytes Received	SCSBytes
sc-bytes	IIS SMTP Bytes Sent	SSCBytes
sc-status	IIS SMTP Successful Protocol Status 2xx	SIIS-2
sc-status	IIS SMTP Redirection Protocol Status 3xx	SIIS-3
sc-status	IIS SMTP Client Error Protocol Status 4xx	SIIS-4
sc-status	IIS SMTP Server Error Protocol Status 5xx	SIIS-5
time-taken	IIS SMTP Time Taken	STimeTkn
Web Resources		
cs-bytes	IIS Web Bytes Received	WCSBytes
sc-bytes	IIS Web Bytes Sent	WSCBytes
sc-status	IIS Web Successful Protocol Status 2xx	WIIS-2
sc-status	IIS Web Redirection Protocol Status 3xx	WIIS-3
sc-status	IIS Web Client Error Protocol Status 4xx	WIIS-4
sc-status	IIS Web Server Error Protocol Status 5xx	WIIS-5
time-taken	IIS Web Time Taken	WTimeTkn

Table 6-2 • Default IIS Identifiers and Resources (Continued)

Setting Up the IIS Collector

This section provides information about the scripts specific to the IIS collector: `JobMSIIS.wsf` and `MSIIS.wsf`.

If you installed CIMS Data Collectors in the default location, the `MSIIS.wsf` script is in `C:\Program Files\CIMSLab\Collectors\MSIIS`.

The `JobMSIIS.wsf` script is in `Processes\MSIIS-Web` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

For information about other scripts related to the collection process, see *Setting Up the System (System Architecture)* on page 2-3.

Setting Up the JobMSIIS.wsf Script

For flexibility, CIMS Lab provides a sample job process script named `JobMSIIS-Web.wsf` that you can modify for any type of IIS collection: Web site, FTP site, or SMTP server (see *Copying the MSIIS-Web Process Definition Folder* on page 6-9).

This script calls and passes parameters to the `MSIIS.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `MSIIS.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the MSIIS.wsf Script

The `MSIIS.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 6-3 • MSIIS Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the MSIIS.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.
ProcessType	The IIS processing type: Web, FTP, or SMTP.
SiteIDOrAll (optional)	This parameter specifies data collection from a particular site or sites or all sites. Enter a site or sites or All. If this parameter is not included, the default is All.

Table 6-3 • MSIIS Script Parameters (Continued)

Copying the MSIIS-Web Process Definition Folder

If you are collecting log files from an IIS type other than Web or you are collecting from more than one IIS type, you can copy the MSIIS-Web process definition folder and modify it for use by other IIS types (FTP or SMTP).

To create a new process definition folder, copy and rename the MSIIS-Web folder and then rename and modify the JobMSIIS-Web.wsf script. The job script name should begin with Job followed by the name of the process definition folder. For example, if the folder name is MSIIS-FTP, the name of the job script should be JobMSIIS-FTP.wsf.

Running the IIS Collector

To run the IIS collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the IIS collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("MSIIS-Web")
```

Note that `Web` could be `FTP` or `SMTP` depending on the IIS type.

Microsoft Internet Security and Acceleration (ISA) Server Data Collector

The CIMS Data Collector for Microsoft ISA Server collects data that is contained in a log files produced by ISA Server. This log file provides useful metrics such as the number of bytes received from and sent to a remote computer and the total time taken to process a request.

The following sections provide instructions for enabling logging for ISA Server and for setting up and running the ISA Server collector.

Enabling ISA Logging

The following provides an example of enabling logging for ISA Server 2000:

To configure logging to a file:

1. In the ISA Management window, click **Logs**.
2. In the details pane, right-click the applicable service, and then click **Properties**.
3. On the **Log** tab, click **File**.
4. Click **Options**.
5. Select the **Compress log files** check box.

To specify fields to log:

1. In the ISA Management window, click **Logs**.
2. On the details pane, right-click the applicable service, and then click **Properties**.
3. On the **Fields** tab, do one of the following:
   - To select specific fields, select the appropriate check box.
   - To clear all of the check boxes in the field list, click **Clear All**.
   - To select all of the check boxes in the field list, click **Select All**.
   - To select a default set of fields in the ISA Server log file, click **Restore Defaults**.

For more information about ISA logging, refer to the Microsoft documentation, including the Microsoft Knowledge Base Article—302372.

ISA Server Log File Format

The fields that appear in the ISA Server log file depend on the fields selected when configuring logging (see *To specify fields to log:* on page 6-11). The following table describes all of the possible record fields in the ISA Server log file. The field names (noted in parentheses) appear when you use the W3C extended log file format.

Field Name	Description/Values
Client IP (`c-ip`)	The IP address of the requesting client.
Client User Name (`cs-username`)	The Windows 2000 logon account name of the user making the request. If ISA Server Access Control is not being used, ISA Server uses `anonymous`.
Client User Agent (`c-agent`)	The client application type sent by the client in the HTTP header. When ISA Server is actively caching, the client agent is `ISA Server`. For the Firewall service, this field includes information about the client's operating system.
Authentication Status (`sc-authenticated`)	Indicates whether or not client has been authenticated with ISA Server. Possible values are `Y` and `N`.
Log Date (`date`)	The date that the logged event occurred.
Log Time (`time`)	The time that the logged event occurred.
Service Name (`s-svcname`)	The name of the service that is logged: ■ `w3proxy` indicates outgoing Web requests to the Web Proxy service. ■ `fwsrv` indicates Firewall service. ■ `w3reverseproxy` indicates incoming Web requests to the Web Proxy service.
Proxy Name (`s-computername`)	The name of the computer running ISA Server. This is the computer name that is assigned in Windows 2000.
Referring Server Name (`cs-referred`)	If ISA Server is used upstream in a chained configuration, this indicates the server name of the downstream server that sent the request.
Destination Name (`r-host`)	The domain name for the remote computer that provides service to the current connection. For the Web Proxy service, a hyphen (`-`) in this field may indicate that an object was retrieved from the Web Proxy server cache and not from the destination.

Table 6-4 • ISA Server Log File Format

Field Name	Description/Values
Destination IP (`r-ip`)	The network IP address for the remote computer that provides service to the current connection. For the Web Proxy service, a hyphen (`-`) in this field may indicate that an object was sourced from the Web Proxy server cache and not from the destination. One exception is negative caching. In that case, this field indicates a destination IP address for which a negative-cached object was returned.
Destination Port (`r-port`)	The reserved port number on the remote computer that provides service to the current connection. This is used by the client application initiating the request.
Processing Time (`time-taken`)	This indicates the total time, in milliseconds, that is needed by ISA Server to process the current connection. It measures elapsed server time from the time that the server first received the request to the time when final processing occurred on the server—when results were returned to the client and the connection was closed. For cache requests that were processed through the Web Proxy service, processing time measures the elapsed server time needed to fully process a client request and return an object from the server's cache to the client.
Bytes Sent (`cs-bytes`)	The number of bytes sent from the client to the server during the current connection. A hyphen (`-`) or a zero or negative number in this field indicates that this information was not provided by the remote computer, or that no bytes were sent to the remote computer.
Bytes Received (`sc-bytes`)	The number of bytes sent from the server and received by the client during the current connection. A hyphen (–) or a zero or negative number in this field indicates that this information was not provided by the remote computer, or that no bytes were received from the server.
Protocol Name (`cs-protocol`)	Specifies the application protocol used for the connection. Common values are `HTTP`, `FTP`, `Gopher`, and `HTTPS` (Secure Hypertext Transfer Protocol). For Firewall service, the port number is also logged.
Transport (`cs-transport`)	Specifies the transport protocol used for the connection. Common values are `TCP` and `UDP`.

Table 6-4 • ISA Server Log File Format (Continued)

Field Name	Description/Values
Operation (s-operation)	Specifies the application method used. For the Web Proxy service, common values are GET, PUT, POST, and HEAD. For the Firewall service, common values are CONNECT, BIND, SEND, RECEIVE, GHBN (GetHostByName), and GHBA (GetHostByAddress).
Object Name (cs-uri)	For the Web Proxy service, this field shows the contents of the URL request. This field applies only to the Web Proxy service log.
Object MIME (cs-mime-type)	The Multipurpose Internet Mail Extensions (MIME) type for the current object. This field may also contain a hyphen (-) to indicate that this field is not used or that a valid MIME type was not defined or supported by the remote computer. This field applies only to the Web Proxy service log.
Object Source (s-object-source)	Indicates the source that was used to retrieve the current object. This field applies only to the Web Proxy service log.
Result Code (sc-status)	This field can be used to indicate: ■ For values less than 100, a Windows (Win32) error code. ■ For values between 100 and 1,000, an HTTP status code. ■ For values between 10,000 and 11,004, a Winsock error code.
Cache Info (s-cache-info)	This number reflects the cache status of the object, which indicates why the object was or was not cached. This field applies only to the Web Proxy service log.
Rule #1 (rule#1)	The rule that either allowed or denied access to the request.
Rule #2 (rule#2)	The second rule that either allowed or denied access to the request.

Table 6-4 • ISA Server Log File Format (Continued)

Field Name	Description/Values
Session ID (`sessionid`)	Identifies a session's connections. For Firewall clients, each process that connects through the Firewall service initiates a session. For secure network address translation (SecureNAT) clients, a single session is opened for all the connections that originate from the same IP address. This field applies only to the Firewall service log.
Connection ID (`connectionid`)	Identifies entries that belong to the same socket. Outbound TCP usually has two entries for each connection: when the connection is established and when the connection is terminated. UDP usually has two entries for each remote address. This field applies only to the Firewall service log.

Table 6-4 • ISA Server Log File Format (Continued)

Identifiers and Resources Collected from the ISA Server Log File

By default, the following fields in the ISA Server log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the ISA Server processing script, `MSISA.wsf`. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
—	Feed (passed from the ISA job script)	—
Client User Name (`cs-username`)	User	—
Resources		
Processing Time (`time-taken`)	MS ISA Server Time Taken	ISATIME
Bytes Sent (`cs-bytes`)	MS ISA Server Bytes Sent	ISASENT
Bytes Received (`sc-bytes`)	MS ISA Server Bytes Received	ISARECV

Table 6-5 • Default ISA Server Identifiers and Resources

Setting Up the ISA Server Collector

This section provides information about the process definition folder and job script for the ISA Server collector.

This section also provides information about the `MSISA.wsf` script. If you installed CIMS Data Collectors in the default location, the `MSISA.wsf` script is in `C:\Program Files\CIMSLab\Collectors\MSISA`.

For information about other scripts related to the collection process, the *Scripts Architecture* section beginning on page 2-13.

Creating a Process Definition Folder and Job Script for the ISA Server Collector

You need to create a process definition folder and script for the ISA Server collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `MSISA`, the name of the job script should be `JobMSISA.wsf`.

The job script for ISA Server calls and passes parameters to the `MSISA.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `MSISA.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13 .

Setting the Parameters for the MSISA.wsf Script

The `MSISA.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 6-6 • MSISA Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the MSISA.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 6-6 • MSISA Script Parameters

Running the ISA Server Collector

To run the ISA Server collector, use the Nightly.bat program (this is shipped as SampleNightly.bat, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the ISA Server collector must be included in the list of jobs in the Nightly.wsf script (this is shipped as SampleNightly.wsf, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("MSISA")
```

Where MSISA is an example process definition folder name.

Microsoft Proxy Server Data Collector

The CIMS Data Collector for Microsoft Proxy Server collects data that is contained in a log files produced by Proxy Server. This log file provides useful metrics such as the number of bytes received from and sent to a remote computer and the total time taken to process a request.

The following sections provide instructions for enabling logging for Proxy Server and for setting up and running the Proxy Server collector.

Enabling Proxy Server Logging

The following are instructions for enabling logging for the Proxy Server 2.0 Web Proxy service on the Windows NT Server 4.0 operating system. You can also follow these instructions to enable logging for the WinSock Proxy and Socks Proxy services. Refer to the Microsoft documentation for instructions on how to enable logging on other platforms.

1. In the Microsoft Management Console window, right-click **Web Proxy**.
2. Click **Properties**. The Web Proxy Service Properties dialog box appears.
3. On the **Logging** tab, select the **Enable logging using** check box and set the general properties such as the log format (regular or verbose), schedule (daily, weekly, monthly, etc.), and location.
4. On the **Permissions** tab, select the **Enable access control** check box and click **WWW** in the **Protocol** list. Enabling access control causes the NT User ID to be added to each record of the log file.
5. Click **OK** to save the settings and close the dialog box.

For more information about Proxy Server logging, refer to the Microsoft documentation.

Proxy Server Log File Format

The following table describes the record fields in the Proxy Server log file.

Field Name	Description/Values	Regular Logging	Verbose Logging
Client IP	The IP address of the requesting client. In cases where active caching is occurring, this field is the same as the Proxy Name field.	X	X
Client User Name	The Windows NT logon account name of the user making the request.	X	X
Client Agent	For the Web Proxy service, indicates specialized header information from the client browser to use when processing the proxy request. For the WinSock Proxy service, indicates the name of the client application that is generating the Windows Socket process request.		X
Client Platform	For the Web Proxy service, this field is not used. For the WinSock Proxy service, indicates the client operating system.		X
Authentication Status	Indicates whether or not the service request is using an authenticated client connection to Proxy Server. Possible values are `Y` and `N`.		X
Log Date	The date that the logged event occurred.	X	X
Log Time	The time that the logged event occurred.	X	X
Service Name	The name of the active service being logged: ■ `CERNProxy` indicates Web Proxy service logging. ■ `WSProxy` indicates WinSock Proxy service logging. ■ `SOCKS` indicates Socks Proxy service logging.	X	X

Table 6-7 • Proxy Server Log File Format

Field Name	Description/Values	Regular Logging	Verbose Logging
Proxy Name	The name of the computer running Proxy Server. This is the name assigned in Windows NT Server 4.0 for the computer.		X
Referring Server Name	If Proxy Server is used upstream in a chained configuration, this indicates the server name of the downstream server that sent the request.		X
Destination Name	The domain name for the remote computer that provides service to the current connection. For the Web Proxy service, a hyphen (-) in this field may indicate that an object was sourced from the Web Proxy Server cache and not from the destination. One exception is negative caching. In that case, this field indicates a destination name for which a negative cached object was returned.	X	X
Destination IP	The network IP address for the remote computer that provides service to the current connection. For the Web Proxy service, a hyphen (-) in this field may indicate that an object was sourced from the Web Proxy Server cache and not from the destination. One exception is negative caching. In that case, this field indicates a destination IP address for which a negative cached object was returned.		X
Destination Port	The reserved port number on the remote computer that provides service to the current connection. This is used by the client application initiating the request.	X	X

Table 6-7 • Proxy Server Log File Format (Continued)

Field Name	Description/Values	Regular Logging	Verbose Logging
Processing Time	This indicates the total time, in milliseconds, that is needed by Proxy Server to process the current connection. It measures elapsed server time from when the server first received the request to the time when final processing occurred on the server—when results were returned to the client and the connection was closed. For cache requests processed through the Web Proxy service, processing time measures the elapsed server time needed to fully process a client request and return an object from the server's cache to the client. For the WinSock Proxy service, the number of bytes received when a connection is terminated. This is in addition to the log generated when the request is processed.		X
Bytes Sent	For the Web Proxy service, the number of bytes sent from the client to the server during the current connection. A hyphen (-) or a zero or negative number in this field indicates that this information was not provided by the remote computer, or that no bytes were sent to the remote computer. For the WinSock Proxy service, the number of bytes sent when a connection is terminated. This is in addition to the log generated when the request is processed.		X
Bytes Received	For the Web Proxy service, the number of bytes sent from the server and received by the client during the current connection. A hyphen (-) or a zero or negative number in this field indicates that this information was not provided by the remote computer, or that no bytes were received from the server.		X

Table 6-7 • Proxy Server Log File Format (Continued)

Field Name	Description/Values	Regular Logging	Verbose Logging
Protocol Name	For the Web Proxy service, specifies the protocol used for transfer (such as `HTTP`, `FTP`, or `Gopher`). For the WinSock Proxy service, specifies a well-known destination port number for the socketed application (such as port `1070` for RealAudio).	X	X
Transport	For the Web Proxy Service, this is always `TCP/IP`. For the WinSock Proxy service, this can be `TCP/IP`, `UDP`, or `IPX/SPX`.		X
Operation	For the Web Proxy service, specifies the current HTTP method used. Possible values are `GET`, `PUT`, `POST`, and `HEAD`. For the WinSock Proxy service, specifies the current socket API call in use. Possible values include `Connect( )`, `Accept( )`, `SendTo( )`, `RecvFrom( )`, `GetHostByName( )`, and `Listen( )`.		X
Object Name (Uri)	For the Web Proxy service, this field shows the contents of the URL request.	X	X
Object MIME	For the Web Proxy service, the Multipurpose Internet Mail Extensions (MIME) type for the current object. This field may also be contain a hyphen (-) to indicate that this field is not used, or that a valid MIME type was not defined or supported by the remote computer.		X

Table 6-7 • Proxy Server Log File Format (Continued)

Field Name	Description/Values	Regular Logging	Verbose Logging
Object Source	For the Web Proxy service, indicates the source used to retrieve the current object.	X	X
Result Code	For the Web Proxy service, this field can be used to indicate: ■ For values less than `100`, a Windows (Win32) error code. ■ For values between `100` and `1,000`, an HTTP status code. ■ For values between `10,000` and `11,004`, a Winsock error code.	X	X

Table 6-7 • Proxy Server Log File Format (Continued)

Identifiers and Resources Collected from the Proxy Server Log File

By default, the following fields in the Proxy Server log file are defined as the chargeback identifiers and resources in the `DefineIdentifier` and `DefineResource` methods in the Proxy Server processing script, `MSProxy.wsf`. The rate codes assigned to the resources *are not* pre-loaded in the CIMS Rate table and must be added to the table as described in the *CIMS Server Administrator's Guide*.

Log File Field	Identifier Name	Rate Code
Identifiers		
—	Feed (passed from the Proxy Server job script)	—
Client User Name (`cs-username`)	User	—
Resources		
Processing Time	—	PROXTIME
Bytes Sent	—	PROXSENT
Bytes Received	—	PROXRECV

Table 6-8 • Default Proxy Server Identifiers and Resources

Setting Up the Proxy Server Collector

This section provides information about the process definition folder and job script for the Proxy Server collector.

This section also provides information about the `MSProxy.wsf` script. If you installed CIMS Data Collectors in the default location, the `MSProxy.wsf` script is in `C:\Program Files\CIMSLab\Collectors\MSProxy`.

For information about other scripts related to the collection process, see the *Scripts Architecture* section beginning on page 2-13.

Creating a Process Definition Folder and Job Script for the Proxy Server Collector

You need to create a process definition folder and script for the Proxy Server collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `MSProxy`, the name of the job script should be `JobMSProxy.wsf`.

The job script for Proxy Server calls and passes parameters to the `MSProxy.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `MSProxy.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13 .

Setting the Parameters for the MSProxy.wsf Script

The `MSProxy.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 6-9 • MSProxy Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the MSProxy.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder specified by the OutputFolder parameter. This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 6-9 • MSProxy Script Parameters (Continued)

Running the Proxy Server Collector

To run the Proxy Server collector, use the Nightly.bat program (this is shipped as SampleNightly.bat, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Proxy Server collector must be included in the list of jobs in the Nightly.wsf script (this is shipped as SampleNightly.wsf, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("MSProxy")
```

Where MSProxy is an example process definition folder name.

SQUID Data Collector

The CIMS Data Collector for SQUID collects data that is contained in a log file produced by SQUID. This log file provides useful metrics such as the number of bytes received from and sent to remote computer and the total time taken to process a request.

Identifiers and Resources Collected from the SQUID Log File

By default, the following field values in the log file are defined as the chargeback identifiers and resource rate codes in the `DefineIdentifier` and `DefineResource` methods in the SQUID processing script, `SQUID.wsf`. The rate codes assigned to the resources *are not* pre-loaded in the CIMS Rate table and must be added to the table as described in the *CIMS Server Administrator's Guide*.

Identifiers

- Feed (this is passed from the job script for the collector)
- ClientIP

Resource Rate Codes

- SQUIDSNT (bytes sent)
- SQUIDRCV (bytes received)

Setting Up the SQUID Collector

This section provides information about the process definition folder and job script for the SQUID collector.

This section also provides information about the `squid.wsf` script. If you installed CIMS Data Collectors in the default location, the `squid.wsf` script is in `C:\Program Files\CIMSLab\Collectors\Squid`.

For information about other scripts related to the collection process, see *Setting Up the System (System Architecture)* on page 2-3.

Creating a Process Definition Folder and Job Script for the SQUID Collector

You need to create a process definition folder and script for the SQUID collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `SQUID`, the name of the job script should be `JobSQUID.wsf`.

The job script for SQUID calls and passes parameters to the `squid.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the squid.wsf script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the squid.wsf Script

The squid.wsf script requires the parameters shown in the following table.

Parameter	Description/Values
LogDate	This parameter specifies the date the log file(s) was created. The values are: ■ preday (previous day) ■ premon (previous month) ■ rndate (current day) ■ curday (current day and previous day) ■ curmon (current month) ■ date in yyyymmdd format This parameter is entered in the command line when running the collector.
RetentionFlag	This parameter is for future use.
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the squid.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 6-10 • Squid Script Parameters

Running the SQUID Collector

To run the SQUID collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the SQUID collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("SQUID")
```

Where `SQUID` is an example process definition folder name.

Sendmail Data Collector

The CIMS Data Collector for sendmail collects data that is contained in a log file produced by sendmail. This log file provides useful metrics such as the number of e-mail messages and bytes received from and sent to a remote computer.

Identifiers and Resources Collected from the Sendmail Log File

By default, the the following field values in the log file are defined as the chargeback identifiers and resource rate codes in the `DefineIdentifier` and `DefineResource` methods in the sendmail processing script, `sendmail.wsf`. The rate codes assigned to the resources *are not* pre-loaded in the CIMS Rate table and must be added to the table as described in the *CIMS Server Administrator's Guide*.

Identifiers

- Feed (this is passed from the job script for the collector)
- Email

Resource Rate Codes

- SMEMRCV (e-mail messages received)
- SMBYRCV (bytes received)
- SMEMSNT (e-mail messages sent)
- SMBYSNT (bytes sent)

Setting Up the Sendmail Collector

This section provides information about the process definition folder and job script for the sendmail collector.

This section also provides information about the `sendmail.wsf` script. If you installed CIMS Data Collectors in the default location, the `sendmail.wsf` script is in `C:\Program Files\CIMSLab\Collectors\SendMail`.

For information about other scripts related to the collection process, see *Setting Up the System (System Architecture)* on page 2-3.

Creating a Process Definition Folder and Job Script for the Sendmail Collector

You need to create a process definition folder and script for the sendmail collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `Sendmail`, the name of the job script should be `JobSendmail.wsf`.

The job script for sendmail calls and passes parameters to the `sendmail.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `sendmail.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the sendmail.wsf Script

The `sendmail.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	This parameter specifies the date the log file(s) was created. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.

Table 6-11 • Sendmail Script Parameters

Parameter	Description/Values
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the sendmail.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 6-11 • Sendmail Script Parameters (Continued)

Running the Sendmail Collector

To run the sendmail collector, use the Nightly.bat program (this is shipped as SampleNightly.bat, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the sendmail collector must be included in the list of jobs in the Nightly.wsf script (this is shipped as SampleNightly.wsf, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("Sendmail")
```

Where Sendmail is an example process definition folder name.

Apache Data Collector

The CIMS Data Collector for Apache collects data that is contained in a log file produced by Apache. This log file provides useful metrics such as the number of Web server hits and the number of bytes transferred from the Web server.

Identifiers and Resources Collected from the Apache Log File

By default, the following field values in the log file are defined as the chargeback identifiers and resource rate codes in the `DefineIdentifier` and `DefineResource` methods in the Apache processing script, `Apache.wsf`. The rate codes assigned to the resources *are not* pre-loaded in the CIMS Rate table and must be added to the table as described in the *CIMS Server Administrator's Guide*.

Identifiers

- Feed (this is passed from the job script for the collector)
- RemoteHost
- User
- AuthUser

Resource Rate Codes

- APHITS (Apache total hits)
- APBYTES (bytes transferred)

Setting Up the Apache Collector

This section provides information about the process definition folder and job script for the Apache collector.

This section also provides information about the `Apache.wsf` script. If you installed CIMS Data Collectors in the default location, the `Apache.wsf` script is in `C:\Program Files\CIMSLab\Collectors\Apache`.

For information about other scripts related to the collection process, see *Setting Up the System (System Architecture)* on page 2-3.

Creating a Process Definition Folder and Job Script for the Apache Collector

You need to create a process definition folder and script for the Apache collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing process definition folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `Apache`, the name of the job script should be `JobApache.wsf`.

The job script for Apache calls and passes parameters to the Apache.wsf, Scan.wsf, ProcCIMS.wsf, and CleanUp.wsf scripts.

The parameters required for the Apache.wsf script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13 .

Setting the Parameters for the Apache.wsf Script

The Apache.wsf script requires the parameters shown in the following table.

Parameter	Description/Values
LogDate	This parameter specifies the date the log file(s) was created. The values are: ■ preday (previous day) ■ premon (previous month) ■ rndate (current day) ■ curday (current day and previous day) ■ curmon (current month) ■ date in yyyymmdd format This parameter is entered in the command line when running the collector.
RetentionFlag	This parameter is for future use.
Feed	The name of the server that contains the log files that you want to process or SELF if the log files are on the same server as the Apache.wsf script. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. The use of a UNC path for the log folder location is recommended.

Table 6-12 • Apache Script Parameters

Running the Apache Collector

To run the Apache collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Apache collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("Apache")
```

Where `Apache` is an example process definition folder name.

Netscape Proxy Server Data Collector

CIMS Lab provides a CIMS Data Collector for Netscape Proxy Server. For instructions on how to configure this collector, contact CIMS Lab (*Chapter 13, Contacting Technical Support*).

7

Storage Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for disk storage. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Windows Disk Data Collector

Note • This collector requires the Microsoft .NET Framework. You can download or order the .NET Framework from Microsoft.

The CIMS Windows Disk collector scans a directory tree and provides a snapshot of the amount of disk space used by each top level folder within a specified drive or folder and the number of files within each folder (including all subfolders).

This collector does not require a usage metering file to produce CIMS Server Resource (CSR) files. The files are produced by the collector's executable program, `CIMSWinDisk.exe`. If you installed CIMS Server in the default location, this program is in `C:\Program Files\CIMSLab\Collectors\CIMSWinDisk`.

The following sections provide instructions for setting up and running the CIMS Windows Disk collector.

Setting Up the CIMS Windows Disk Collector

The CIMS Windows Disk collector includes an XML file (`CIMSWinDisk.xml`) that contains the parameters required to run the collector executable program and scripts. You need to edit the parameter values in this file for your site.

Because the `CIMSWinDisk.xml` file contains the required parameters for the collector, you do not need to edit the job script (`JobCIMSWinDisk.wsf`) unless you want to do any of the following:

- Change the values for optional parameters. The `JobCIMSWinDisk.wsf` script passes optional parameters to the `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts. For a description of optional parameters by script, see the *Scripts Architecture* section beginning on page 2-13.
- Exclude an instance or instances in the XML file from processing. For more information, see *About the CIMSWinDisk.xml File* on page 7-3.

If you installed CIMS Data Collectors in the default location, the `CIMSWinDisk.xml`file and `JobCIMSWinDisk.wsf` script are in `Processes\CIMSWinDisk` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

About the CIMSWinDisk.xml File

The `JobCIMSWinDisk.wsf` script passes the following parameters to the executable program for the CIMS Windows Disk collector:

- The location of the XML file
- The collector name
- The collector instance name (optional)

> **Note • By default, the collector instance name is not included in the `JobCIMSWinDisk.wsf` script. This parameter would be required only if you wanted to process only one of multiple instances. An alternate way to exclude collector instances is to use the `Active` parameter as described on** page 7-4.

The collector name and instance name (if applicable) are matched to the corresponding names in the `CIMSWinDisk.xml` file. The file groups parameters by collector and instance as follows:

```
<Collector name="CIMSWinDisk" instanceName="CIMSLAB-C" instanceDescription="Scan of
CIMSLAB C" Active="True">
    <Parameters>
        <Parameter name="LogDate" value="PREDAY" />
        <Parameter name="Retention" value="KEEP" />
        <Parameter name="Feed" value="CIMSLAB-C" />
        <Parameter name="OutputFolder" value="C:\Program Files\CIMSLab\Sample
         Processes\CIMSWinDisk" />
        <Parameter name="PathToScan" value="C:\" />
        <Parameter name="Units" value="GB"
        <Parameter name="NumberOfLevels" value="1" /
    </Parameters>
</Collector>
```

By default, the `CIMSWinDisk.xml` file contains two instances of the CIMS Windows Disk collector: `CIMSLAB-C` (shown previously) and `CIMSLAB-D`. You can edit the file to include as many instances of the collector as needed to collect data from multiple drives and/or folders.

Editing the CIMSWinDisk.xml File

The following table describes the parameters in the `CIMSWinDisk.xml`file. All parameters are required unless noted. Edit these parameters as needed.

Parameter	Description/Values
`Collector Name`	The collector name. *Do not change this parameter*.
`instanceName`	The name of the instance for the collector.
`instanceDescription`	A description of the instance for the collector.
`Active`	If this parameter is set to `True`, the instance is included in processing. If this parameter is set to `False`, the instance is excluded from processing.
`LogDate`	Regardless of the date value entered for this parameter (`PREDAY`, `CURDAY`, etc.), the data collected by CIMS Windows Disk reflects the date and time that the collector was run. However, the date in file name for the CSR file and the start and end date in the CSR records reflect this parameter value. For example, if the value is `PREDAY`, the previous day's date appears in the file name and start and end dates. **Note:** The log date entered on the command line when running the `Nightly.bat` or `Monthly.bat` script does not affect the parameter value entered here.
`Retention`	This parameter is for future use.
`Feed`	This parameter automatically creates a subfolder of the same name in the process definition folder (see the `OutputFolder` parameter). This subfolder is used to store the CSR files created by CIMS Windows Disk (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
`OutputFolder`	The process definition folder for the collector. This is the location of the final CSR file that is created by the `Scan.wsf` script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.

Table 7-1 • CIMSWinDisk.xml Parameters

Parameter	Description/Values
PathToScan	Valid values for this parameter are: ■ The drive or folder one level above the folder information you want to collect. For example, "PathToScan" value="\\ComputerA\C\" collects data for all top level folders under the C share. Note that \\ComputerA\C\ is an example UNC path, which is recommended. ■ All, to scan the top level folders under all drives with an administrative share (C$ through Z$). Note that only shared drives are scanned when you specify All. **Note:** To scan a shared drive, the Windows user ID used to log on to the computer running CIMSWinDisk must have authority to scan the share.
Units (optional)	If the parameter value is set to GB, is left blank, or is not included, disk space usage is presented in gigabytes. To present the usage units in another measurement, enter one of the following values: ■ bytes ■ KB (kilobytes) ■ MB (megabytes) ■ A number by which you want to divide the usage units. In this case, the units are measured in bytes rather than gigabytes.
NumberOfLevels (Optional)	This parameter works in conjunction with the PathToScan parameter to determine the folder level that will be scanned. For example, if the PathToScan is All (scan all drives) and the NumberOfLevels parameter is 2, the data collection will reflect all second level folders under the scanned drives.

Table 7-1 • CIMSWinDisk.xml Parameters (Continued)

Identifiers and Resources Collected by the CIMS Windows Disk Collector

By default, the CIMS Windows Disk collector creates the following chargeback identifiers and resource rate codes from the data collected. The rate codes are pre-loaded in the CIMS Rate table.

Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers	
Feed (the feed name specified by the `Feed` parameter in `CIMSWinDisk.xml` file)	—
Folder (the folder name determined by the `PathToScan` parameter in `CIMSWinDisk.xml` file)	—
Resources	
MS Windows Disk Folder Usage in GB	DISKFILE
MS Windows Files in Folder	DISKSIZE

Table 7-2 • Default CIMS Windows Disk Identifiers and Resources

Running the CIMS Windows Disk Collector

To run the CIMS Windows Disk collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the CIMS Windows Disk collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("CIMSWinDisk")
```

Disk Directory (DiskDir) Data Collector

Note • This collector is deprecated. CIMS Lab strongly recommends that you use the CIMS Windows Disk data collector (see page 7-2).

The DiskDir collector scans a directory tree and provides a snapshot of the amount of disk space (in megabytes) used by each top level folder within a specified drive or folder and the number of files within each folder. The number of files does not include files in subfolders.

This collector does not require a usage metering file to produce CSR files.

The following sections provide instructions for setting up and running the DiskDir collector.

Setting Up the DiskDir Collector

This section provides information about creating the process definition and job script for the DiskDir collector.

This section also provides information about the `DiskDir.wsf` script. If you installed CIMS Data Collectors in the default location, the `DiskDir.wsf` script is in `C:\Program Files\CIMSLab\Collectors\DiskDir`.

For information about other scripts related to the collection process, see the *Scripts Architecture* section beginning on page 2-13.

Creating a Job Folder and Job Script for DiskDir Data Collector

You need to create a job folder and script for the DiskDir collector in the `Processes` folder (see *About the Processes Folder* on page 2-8). For convenience, you can copy and rename any existing job folder that contains a job script and then rename and modify the script. The job script name should begin with `Job` followed by the name of the job folder. For example, if the folder name is `DiskDir`, the name of the job script should be `JobDiskDir.wsf`.

The job script for DiskDir calls and passes parameters to the `DiskDir.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `DiskDir.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13 .

Setting the Parameters for the DiskDir.wsf Script

The `DiskDir.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	Regardless of the date value entered in the command line when running the collector (`preday`, `curday`, etc.), the data collected by DiskDir reflects the date and time that the collector was run.
`RetentionFlag`	This parameter is for future use.
`Feed`	Because the input server is the server running `DiskDir.wsf` script, you can either enter the name of the server or `SELF`. A subfolder with the same name as the server is automatically created in the process definition folder (see the `OutputFolder` parameter). This subfolder is used to store the CSR files created the the DiskDir collector (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
`OutputFolder`	The process definition folder for the collector. This is the location of the final CSR file that is created by the `Scan.wsf` script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
`SourceFolder`	The drive or folder one level above the folder information you wish to collect. For example, `SourceFolder=C` collects data for all top level folders on drive `C`. If you specify `All`, then data is collected for all top level folders on all disk drives on the input server.

Table 7-3 • DiskDir Script Parameters

Identifiers and Resources Collected by the DiskDir Collector

By default, the DiskDir collector creates the following chargeback identifiers and resource rate codes from the data collected. The identifiers and rate codes are defined in the `DefineIdentifier` and `DefineResource` methods in the DiskDir processing script, `DiskDir.wsf`. The rate codes are pre-loaded in the CIMS Rate table.

Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers	
Feed (passed from the DiskDir job script)	—
Server	—
Drive	—
Folder	—
Path	—
Attributes	—
Type	—
ParentFolder1	—
ParentFolder2	—
ParentFolder3	—
Resources	
MS Windows Disk Size in MB	DISKSIZE
MS Windows Number of Files	DISKFILE

Table 7-4 • Default DiskDir Identifiers and Resources

Running the DiskDir Collector

To run the DiskDir collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the DiskDir collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("DiskDir")
```

Where `DiskDir` is an example job folder name.

Veritas

CIMS Lab provides a CIMS Data Collector for Veritas. For instructions on how to configure this collector, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

8

Network Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for network applications. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Novell NetWare Data Collection

To collect usage data for Novell Netware, use the CIMS Windows Disk collector (see *Windows Disk Data Collector* on page 7-2). The CIMS Windows Disk collector scans a directory tree and provides a snapshot of the amount of disk space used by each top level folder within a specified drive or folder and the number of files within each folder (including all subfolders).

BindView Data Collector

CIMS Lab provides a CIMS Data Collector for BindView. For instructions on how to configure this collector, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

9

Printer Data Collectors

This chapter contains instructions for setting up and running CIMS Data Collectors for printers. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

Windows Event Log Data Collector for Print

Note • This collector requires the Microsoft .NET Framework. You can download or order the .NET Framework from Microsoft.

The CIMS Windows Event Log collector gathers printer events from the Windows System event log on a print server or servers. The collector provides useful metrics such as:

- The name of the user that ran the print job.
- The number of pages printed and the print job size in kilobytes.

The following sections provide instructions for setting up and running the CIMS Windows Event Log collector.

Setting Up the CIMS Windows Event Log Collector

The CIMS Windows Event Log collector includes an XML file (`CIMSWinEventLog.xml`) that contains the parameters required to run the collector executable program (`CIMSWinEventLog.exe`) and scripts. You need to edit the parameter values in this file for your site.

Because the `CIMSWinEventLog.xml` file contains the required parameters for the collector, you do not need to edit the job script (`JobCIMSWinEventLog.wsf`) unless you want to do any of the following:

- Change the values for optional parameters. The `JobCIMSWinEventLog.wsf` script passes optional parameters to the `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts. For a description of optional parameters by script, see the *Scripts Architecture* section beginning on page 2-13.
- Exclude an instance or instances in the XML file from processing. For more information, see *About the CIMSWinEventLog.xml File* on page 9-3.

If you installed CIMS Data Collectors in the default location, the `CIMSWinEventLog.xml` file and `JobCIMSWinEventLog.wsf` script are in `Processes\CIMSWinEventLog` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

`CIMSWinEventLog.exe` is in `C:\Program Files\CIMSLab\Collectors\CIMSWinEventLog`.

About the CIMSWinEventLog.xml File

The JobCIMSEventLog.wsf script passes the following parameters to the executable program for the CIMS Windows Event Log collector:

- The location of the XML file
- The collector name
- The collector instance name (optional)

> **Note • By default, the collector instance name is not included in the JobCIMSWinEventLog.wsf script. This parameter would be required only if you wanted to process only one of multiple instances. An alternate way to exclude collector instances is to use the Active parameter as described on** page 9-4**.**

The collector name and instance name (if applicable) are matched to the corresponding names in the CIMSWinEventLog.xml file. The file groups parameters by collector and instance as follows:

```
<Collector name="CIMSWinEventLog" instanceName="ROCA-FILESRVR" instanceDescription=
"ROCA-FILESRVR Print Server" Active="True">
    <Parameters>
        <Parameter name="LogDate" value="PREDAY" />
        <Parameter name="Retention" value="KEEP" />
        <Parameter name="Feed" value="ROCA-FILESRVR" />
        <Parameter name="OutputFolder" value="C:\Program Files\CIMSLab\Sample
         Processes\CIMSWinEventLog" />
        <Parameter name="LogSource" value="ROCA-FILESRVR" />
        <Parameter name="LogType" value="Server"/>
        <Parameter name="EventType" value="Print" />
    </Parameters>
</Collector>
```

By default, the CIMSWinEventLog.xml file contains one instances of the CIMS Windows Event Log collector. You can edit the file to include as many instances of the collector as needed. For example, if you want to collect data from multiple servers.

Editing the CIMSWinEventLog.xml File

The following table describes the parameters in the `CIMSWinEventLog.xml`file. All parameters are required. Edit these parameters as needed.

Parameter	Description/Values
`Collector Name`	The collector name. *Do not change this parameter*.
`instanceName`	The name of the instance for the collector.
`instanceDescription`	A description of the instance for the collector.
`Active`	If this parameter is set to `True`, the instance is included in processing. If this parameter is set to `False`, the instance is excluded from processing.
`LogDate`	This parameter specifies the date that the print events in the Windows System event log occurred. The values are: ■ `preday` (previous day) ■ `prewek` (previous week) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ `curwek` (current week) ■ date in `yyyymmdd` format **Note:** The log date entered on the command line when running the `Nightly.bat` or `Monthly.bat` script does not affect the parameter value entered here.
`Retention`	This parameter is for future use.
`Feed`	The name of the server that contains the event log or `SELF` if the event log is on the same server as the collector. A subfolder with the same name as the server is automatically created in the process definition folder (see the `OutputFolder` parameter). This subfolder is used to store the CIMS Server Resource (CSR) files that are created by the collector (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.

Table 9-1 • CIMSWinEventLog.xml Parameters

Parameter	Description/Values
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogSource	This parameter depends on the log type (see the LogType parameter). If you are collecting events from an archived event log, enter the path and filename of the archived file. Note that the file must be an event log file (.evt). You cannot use logs archived as .txt or .csv files. The use of a UNC path is recommended. If you are collecting events directly from the event log, enter the name of the server that contains the event log or SELF if the event log is on the same server as the collector.
LogType	The type of log. The values are: ■ file (if collecting from archived .evt files) ■ server (if collecting directly from the event log)
EventType	Do not change the default parameter value, Print. This value instructs the collector to gather data from events that are identified by **Print** in the **Source** column of the event log.

Table 9-1 • CIMSWinEventLog.xml Parameters (Continued)

Identifiers and Resources Collected by the CIMS Windows Event Log Collector

By default, the CIMS Windows Event Log collector creates the following chargeback identifiers and resource rate codes from the data collected. The rate codes are pre-loaded in the CIMS Rate table.

Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers	
Feed (the feed name specified by the `Feed` parameter in `CIMSWinEventLog.xml` file)	—
UserName (the name of the user that ran the print job)	—
PrinterName (the name of the printer that produced the print job)	—
JobNumber (a job number assigned by the system)	—
JobName (an application-defined description of the document printed)	—
PortName (the printer port name)	—
Resources	
MS Windows Print Print KBytes	WPRTPRKB
MS Windows Print Page Count	WPRTPRPC

Table 9-2 • Default CIMS Windows Event Log Identifiers and Resources

Running the CIMS Windows Event Log Collector

To run the CIMS Windows Event Log collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the CIMS Windows Event Log collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("CIMSWinEventLog")
```

Setting the Event Viewer Options for the System Event Log

Because the CIMS Windows Event Log collector gathers data from the Windows System event log on the print server, you should set the log size and overwrite options on the print server as described in the following steps. (Note that these steps are for the Microsoft Windows Server 2000 operating system. If you are using another operating system, refer to the Microsoft documentation if needed.)

1. In Windows, click **Start** ▸ **Settings** ▸ **Control Panel**.
2. Double-click **Administrative Tools** ▸ **Event Viewer**.

   The Event Viewer window appears.
3. Right-click **System**, and then click **Properties**.

   The System Properties dialog box appears.
4. On the **General** tab, make sure that the **Maximum log size** is set to a size that will accommodate your collection schedule. For example, you might need to set a larger log size if you are collecting print events on a monthly schedule rather than a daily schedule.
5. Choose one of the following options under **When maximum log size is reached:**

   - Click **Overwrite events older than** and enter a number 30 day longer than your collection schedule. For example, if you are collecting events daily, set the number to 31. If you are collecting events monthly, set the number to 60.
   - Click **Do not overwrite events**. This option requires that you clear the log manually rather than automatically when the log is full.

   To avoid deleting older events, do not click **Overwrite events as needed**.
6. Click **OK** when you are finished.

Windows Print Data Collector

The CIMS Windows Print collector gathers printer usage data for printers connected to a print server and produces a log file of the data (see *CIMS Windows Print Collector Log File Format* on page 9-11). This log file provides useful metrics such as:

- The name of the user that ran the print job.
- Number of pages submitted and printed and the print job size in kilobytes.
- Number of copies printed.

The following sections provide instructions for installing the CIMS Windows Print collector, enabling logging, and setting up and running the collector.

Installing the CIMS Windows Print Collector

You need to install the CIMS Windows Print collector on each print server that you want to collect data from. CIMS Lab provides a simple setup program, `CIMSWinPrintSetup.exe`, for installing the CIMS Windows Print collector on print servers. This setup program includes the following components:

- **The CIMS Windows Print collector.** This installs the following components in the `Collectors\CIMSWinPrint` folder created during installation:
  - The processing script, `CIMSWinPrint.wsf`.
  - The executable program for the collector, `CIMSWinPrintService.exe`.
  - An executable program, `CIMSWinPrintServiceLog.exe`, that is used by CIMS Lab for troubleshooting purposes. For more information about this program, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).
  - The executable program for the collector's administrative program, `CIMSWinPrintServiceAdmin.exe`.
- **CIMS Aggregation Engine (`CIMSAggregation.dll`).** CIMS Aggregation Engine is called by the `CIMSWinPrint.wsf` script. CIMS Aggregation Engine aggregates the records within the print log file by identifier values and produces a CSR file. For more information about CIMS Aggregation Engine, see *Processing Script* **on page 2-5**.
- **Support Files.** These files support the collector's administrative program and are needed only if **CIMS Server Administrator** is not installed on the computer.

This installation does not include CIMS Processing Engine, which processes the CSR files created by CIMS Aggregation Engine and loads the output data into the database. To process CSR files, you need to process the files on the central CIMS Data Collectors server. For more information, see *Setting Up the CIMS Windows Print Collection Process* on page 9-14.

To install the CIMS Windows Print collector:

Note • These following steps are also applicable if you are upgrading to a new version or release of the CIMS Windows Print collector.

1. Log on to Windows as an Administrator.
2. Click the Windows **Start** button, and then click **Run**.
3. Enter the path to the setup program `CIMSWinPrintSetup.exe`, and then click **OK**. (The path depends on the location of the setup program, i.e., CIMS Lab Product CD, network drive, etc. For example, if you are installing from CD, the path might be `D:\CIMSServer\CIMSWinPrintSetup.exe`).
4. In the setup wizard, select the collector components or component that you want to install as follows, and then click **Next**:
   - To install multiple components, click **Custom**, and then select the check boxes for the components that you want to install.
   - To install an individual component, click the component that you want to install.
5. Choose the default location for installation (`C:\Program Files\CIMSLab`) or click **Browse** to choose another location. After making your selection, click **Install**.

Enabling Windows Print Logging

To enable logging for the CIMS Windows Print collector, you need to create an output folder for the log files created by the collector and to configure the collector as described in the following sections.

Creating an Output Folder for Storing Log Files

You need to create an output folder for storing the log files that are created by the CIMS Windows Print collector. This folder must be on the same computer as the collector, not on the central CIMS Data Collectors server. You should create this folder in a location where you keep data that is backed up.

Important! • Do not store log files (`CIMSPrintLog-`*`yyyymmdd`*`.txt`) in the `Processes\CIMSWinPrint\`*`feedname`* folder on the central CIMS Data Collectors server. The feed folder should contain only CSR files.

Configuring the CIMS Windows Print Collector

The CIMS Windows Print collector tracks print jobs for selected printers connected to the print server and enters the usage data for each job as a record in the log file.

The CIMS Windows Print collector includes an easy-to-use GUI program for configuring and enabling the collector. To use this program, click the **Start** menu, and then click **Programs ▸ CIMS Server ▸ Collectors ▸ CIMS Windows Print Administrator** and set the following options:

- **Log file path.** Enter the path to the folder that you created in *Creating an Output Folder for Storing Log Files*. The default is the path where the `CIMSWinPrintSevice.exe` program is located. The use of a UNC path for the file location is recommended.
- **Log file prefix.** The default name for the log file is `CIMSPrintLog-yyyymmdd.txt`. You can use the default prefix `CIMSPrintLog-` or replace it with the prefix of your choice (or no prefix).
- **Use Local Time in output records.** If this check box is selected (the default), the local time set for the computer is used in the date and time fields in the log file. If this check box is cleared, Universal Time Coordinate (*UTC*) time is used in the log file.

  Note • The log file *date* always reflects local time, regardless of whether Use Local Time is selected.

- **Monitored Printer List.** The printers that you want to monitor for data collection. Click Add or Remove to add or delete printers from this list.

  When you click **Add**, the Select Printers dialog box appears. After the collector searches for shared local and network printers, you can do one of the following:

  - Enter a printer name in the **Printer** box.
  - Select one of the printers listed in **Available Printers**.

- **Control Service.** Click this button open the Service Control dialog box to start or stop the CIMS Windows Print collector. You can also start and stop the collector from the Windows Control Panel and then click the **Refresh** button in the Service Control dialog box to make the change in the collector.

CIMS Windows Print Collector Log File Format

The following table describes the record fields in the log file produced by the CIMS Windows Print collector.

Field Name	Description/Values
RecordType	The record type is J for job.
JobID	The job ID assigned by the system.
MachineName	The name of the computer that generated the print job.
UserName	The name of the user that ran the print job.
PrinterName	The name of the printer that produced the print job.
PrinterServerName	The name of the print server for the printer that produced the print job.
PrinterShareName	The share name of the printer that produced the print job.
JobName	An application-defined description of the document printed.
PortName	The printer port name.
SubmitKBytes	The number of kilobytes submitted.
PrintKBytes	The number of kilobytes printed.
SubmitPageCount	The number of pages submitted.
PrintPageCount	The number of pages printed.
Copies	The number of copies printed.
Priority	The priority of the print job in the print server queue.
SubmitDateTime	The date and time that the print job was submitted.
CompleteDateTime	The date and time that the print job was completed.
DuplexType	Indicates whether the print job is single- or double-sided.
FormName	The form name. If there is no form name a hyphen (–) appears.
Orientation	Portrait or landscape.
PrintQuality	The print quality in dots per inch (DPI).

Table 9-3 • CIMS Windows Print Collector Log File Format

Field Name	Description/Values
PaperSource	The source of the paper, for example, automatically select, manual feed, or tray number.
PaperSize	The paper size.
ColorOutput	Specifies whether the print output was in color.

Table 9-3 • CIMS Windows Print Collector Log File Format (Continued)

Identifiers and Resources Collected from the CIMS Windows Print Collector Log File

By default, the following fields in the CIMS Windows Print collector log file are defined as the chargeback identifiers and resources in the DefineIdentifier and DefineResource methods in the CIMS Windows Print processing script, CIMSWinPrint.wsf. The rate codes assigned to the resources are pre-loaded in the CIMS Rate table.

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
—	Feed (passed from the JobCIMSWinPrint script)	—
MachineName	MachineName	—
UserName	User	—
PrinterName	PrinterName	—
PrinterServerName	Server	—
PrinterShareName	PrinterShareName	—
JobName	JobName	—
PortName	PortName	—
Priority	Priority	—
DuplexType	DuplexType	—
FormName	FormName	—
Orientation	Orientation	—
PrintQuality	PrintQuality	—
PaperSource	PaperSource	—

Table 9-4 • Default CIMS Windows Print Identifiers and Resources

Log File Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
ColorOutput	ColorOutput	—
Resources		
SubmitKBytes	MS Windows Print Submit KBytes	WPRTSBKB
PrintKBytes	MS Windows Print Print KBytes	WPRTPRKB
SubmitPageCount	MS Windows Print Submit Page Count	WPRTSBPC
PrintPageCount	MS Windows Print Page Count	WPRTPRPC
Copies	MS Windows Print Copies	WPRTCOPY

Table 9-4 • Default CIMS Windows Print Identifiers and Resources (Continued)

Setting Up the CIMS Windows Print Collection Process

The following sections provide steps for setting up the CIMS Windows Print collection process. These steps differ depending on whether you are processing the log files produced by the CIMS Windows Print collector on the central CIMS Data Collectors server or whether you are processing the log files on the computer running the CIMS Windows Print collector.

Note • Although you can process log files on the computer running the CIMS Windows Print **collector, you should not process the resulting CSR files on this computer. You should process CSR files on the CIMS Data Collectors server.**

Of the two options for processing log files, processing the log files on the central CIMS Data Collectors server is the simpler option. However, if the log files are large, you should have a quicker elapsed completion time if you process the files on the computer running the CIMS Windows Print collector.

Option 1—To process the log files on the central CIMS Data Collectors server:

On the central CIMS Data Collectors server, set up the collector and related scripts as described in *Setting Up the System (System Architecture)* on page 2-3. The job script for the collector is `JobCIMSWinPrint.wsf`. The processing script for the collector is `CIMSWinPrint.wsf`.

If you installed the CIMS Windows Print collector in the default location on the central CIMS Data Collectors server, the `CIMSWinPrint.wsf` script is in `C:\Program Files\CIMSLab\Collectors\CIMSWinPrint`. The `JobCIMSWinPrint.wsf` script is in `Processes\CIMSWinPrint` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

The `JobCIMSWinPrint.wsf` script passes parameters to the `CIMSWinPrint.wsf` script. These parameters are described in *Setting the Parameters for the CIMSWinPrint Script* on page 9-15.

Option 2—To process the log files on a computer running the CIMS Windows Print collector:

1. On the computer running the CIMS Windows Print collector, create a batch script to call the `CIMSWinPrint.wsf` script that is on the same computer. If you installed the CIMS Windows Print collector in the default location on the computer, the `CIMSWinPrint.wsf` script is in `C:\Program Files\CIMSLab\Collectors\CIMSWinPrint`. The parameters required for the `CIMSWinPrint.wsf` script are described in *Setting the Parameters for the CIMSWinPrint Script* on page 9-15.

2. On the CIMS Data Collectors server, set up the collector as described in *Setting Up the System (System Architecture)* on page 2-3. In the `JobCIMSWinPrint.wsf` script, remove the call to the `CIMSWinPrint.wsf` script.

Setting the Parameters for the CIMSWinPrint Script

The CIMSWinPrint.wsf script requires the parameters shown in the following table.

Parameter	Description/Values
LogDate	This parameter specifies the date the log file(s) was created. The values are: ■ preday (previous day) ■ premon (previous month) ■ rndate (current day) ■ curday (current day and previous day) ■ curmon (current month) ■ date in yyyymmdd format This parameter is entered in the command line when running the collector.
RetentionFlag	This parameter is for future use.
Feed	The name of the server that contains the log file folder. Because this folder must reside on the same server as the CIMSWinPrint script, you can either enter the name of the server or SELF. A subfolder with the same name as the server is automatically created in the process definition folder (see the OutputFolder parameter). This subfolder is used to store the CSR files that are created by the collector (see *Feed Subfolder* on page 2-12). This parameter is included as an identifier in the CSR file.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file that is created by the Scan.wsf script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
LogFolder	The location of the log file. This folder must be on the same server as the CIMSWinPrint script.

Running the CIMS Windows Print Collection Process

Run the CIMS Windows Print collection process on the central CIMS Data Collectors server using the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the print collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("CIMSWinPrint")
```

10

Transactions Collector

About Transactions

In some circumstances, you might want to generate a CIMS Server Resource (CSR) file for occurrences that are not contained in a usage metering file. For example, you might want to generate a CSR file to apply a credit for an overcharge or to charge for a one time occurrence such as the cost of providing a computer to a new employee.

In these cases, you can create a miscellaneous, recurring, or credit transaction in CIMS Server Web Reporting. These transactions contain the chargeback information that you want to include in a CSR file. For more information about transactions, refer to the *CIMS Server Web Reporting User's Guide*.

Transactions are stored in the CIMS Transaction table in the CIMS Server database. The Transactions collector is used to collect, convert, and process the transactions on a monthly basis.

The following sections provide instructions for setting up and running the Transactions collector.

About the CIMS Transaction Table

The CIMS Transaction table contains the following fields.

Field Name	Field Description
`TransactionUID`	The unique identifier for the transaction.
`AccountCode`	The account code for the transaction.
`TransactionType`	The transaction type: ■ `M` (Miscellaneous ■ `R` (Recurring) ■ `C` (Credit)
`ShiftCode`	The shift code for the transaction.
`RateCode`	The rate code for the transaction.
`ResourceAmount`	The amount of the transaction.

Table 10-1 • CIMS Transaction Table Fields

Field Name	Field Description
Frequency1	Applicable only to recurring transactions. The frequency that the transaction should occur (every month, every 6 months, etc.). Frequency is based on the calendar year (January–December) ■ 1 (monthly) ■ 2 (every other month) ■ 3 (every quarter) ■ 4 (every four months) ■ 6 (every six months) ■ 12 (once a year)
Frequency2	Applicable only to recurring transactions. The period in which the transaction should be processed. This value corresponds to the value in the Frequency1 field. For example, if the value in the Frequency1 field is 6, a value of 1 in this field indicates the first month of a 6 month period (January or July).
FromDate/ToDate	Applicable only to miscellaneous and credit transactions. The date range that the transaction is to occurr.
DateTimeSent	The date and time that the transaction was exported to a flat file.
DateTimeModified	The date and time that the transaction was last modified.
DateTimeEntered	The date and time that the transaction was created.
DateTimeStartProcessing	Applicable only to recurring transactions. The first day that the transaction will be processed.
DateTimeStopProcessing	Applicable only to recurring transactions. The last day that the transaction will be processed.
UserID	The CIMS Server Web Reporting user ID of the person who entered the transaction.
Note	A description of the transaction.
DateTimeDeleted	The date and time that the transaction was deleted.

Table 10-1 • CIMS Transaction Table Fields (Continued)

Identifiers and Resources Collected from the CIMS Transaction Table

By default, the Transactions collector creates the following chargeback identifiers and resource rate codes from the transactions collected. The rate codes are pre-loaded in the CIMS Rate table.

CIMS Transaction Table Field	Identifier Name or Resource Description in CIMS Server	Assigned Rate Code in CIMS Server
Identifiers		
AccountCode	Account_Code	—
Note	Description	—
DateTimeSent	DateTimeSent	—
DateTimeModified	DateTimeModified	—
DateTimeEntered	DateTimeEntered	—
UserID	UserID	—
Resources		
RateCode	This is the rate code assigned to the transaction. It can be any rate code defined in the CIMS Rate table.	Same

Table 10-2 • Default Transaction Identifiers and Resources

Setting Up the Transaction Collector

This section provides information about the scripts specific to the transaction collector: `JobTransactions.wsf` and `Transactions.wsf`.

If you installed CIMS Data Collectors in the default location, the script `Transactions.wsf` is in `C:\Program Files\CIMSLab\Collectors\Transactions`.

The script `JobTransactions.wsf` is in `Processes\Transactions` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8).

For information about the general scripts required by the transaction collector, see the *Scripts Architecture* section beginning on page 2-13.

Setting Up the JobTransactions.wsf Script

The JobTransactions.wsf script calls and passes parameters to the Transactions.wsf, Scan.wsf, ProcCIMS.wsf, and CleanUp.wsf scripts.

The parameters required for the Transactions.wsf script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the Transactions.wsf Script

The Transactions.wsf script requires the parameters shown in the following table.

Parameter	Description/Values
LogDate	This parameter specifies the date the log file(s) was created. The values are: ■ premon (previous month) ■ curmon (current month) This parameter is entered in the command line when running the collector.
RetentionFlag	This parameter is for future use.
OutputFolder	The process definition folder for the collector. This is the location of the final CSR file. The file name for the resource file is changed from CurrentCSR.txt to CurrentCSRPrevious.txt after processing. For more information about the process definition folder, see *Processes Architecture*.
DataSourceID	The ODBC data source UID for the CIMS Server database (see page 2-10 for a description of this parameter).

Table 10-3 • Transaction Script Parameters

Parameter	Description/Values
`Period or Keyword`	The date that the transaction is to occur. This date was specified when the transaction was created. The values are: ■ **Current.** All transactions with dates for the current month will be converted. ■ **Previous.** All transactions with dates for the previous month will be converted. ■ **Period.** All transactions with dates for the specified period (1–13) will be converted. The period is dependent on the type of transaction (that is, miscellaneous, recurring, or credit). This parameter requires the `Year` parameter.
`Year (optional)`	This parameter is required if a period number rather than a keyword is used as a date parameter.

Table 10-3 • Transaction Script Parameters (Continued)

Running the Transactions Collector

To run the Transactions collector, use the `Monthly.bat` program (this is shipped as `SampleMonthly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Transactons collector must be included in the list of jobs in the `Monthly.wsf` script (this is shipped as `SampleMonthly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("Transactions")
```

11

Other Data Collectors

SAP, Shiva, and Evolve Data Collectors

CIMS Lab provides a CIMS Data Collector for SAP, Shiva, and Evolve. For instructions on how to configure these collectors, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

12

CIMS Universal Data Collector

This chapter describes how to use the CIMS Universal Data Collector. You should have a good understanding of the CIMS Data Collector system architecture and have modified the appropriate files as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3 before continuing with the collector-specific information in this chapter.

About CIMS Universal Data Collector

In addition to the data collectors described in the preceding chapters, CIMS Lab provides a universal data collector, CIMS Universal Data Collector, for applications that do not have a specific CIMS Data Collector. The Universal collector uses the CIMS Conversion Engine utility to convert *any usage metering data from any application* to a CIMS Server Resource (CSR) file.

The Data Conversion Process

The following are the files and CIMS components used in the data conversion process.

Application Usage Metering File

The usage metering file can be either of the following:

- Any ASCII file with either fixed length fields or delimited fields (for example, a log file). Each file entry must be on a single line.
- Any database log file.

Conversion Definition

The conversion definition is a file that defines the format of the usage metering file as well as the data that will appear in the output CSR file. You can create conversion definitions using the CIMS Conversion Builder application (see *Creating a Definition file* on page 12-3) or you can create definition files using a text editor such as Notepad (see *Conversion Definition Viewed in Notepad* on page 12-37).

If you have multiple usage metering files with different formats, you need to create a separate conversion definition for each file type. Each conversion definition should be stored in a separate process definition folder and requires a separate job script as described in *Setting Up the Universal Collector* on page 12-25.

Universal Collector

The Universal collector calls the CIMS Conversion Engine (`ProcConvEng.wsf`) to convert the usage metering file. The steps required to set up and run the Universal collector are provided in *Setting Up and Running the Universal Collector* on page 12-25.

CIMS Conversion Engine

CIMS Conversion Engine converts the data in the usage metering file based on the conversion definition. The output form CIMS Conversion Engine is a CSR file.

Creating a Conversion Definition Using CIMS Conversion Builder

You can create a conversion definition as a text file (see *Conversion Definition Viewed in Notepad* on page 12-37); however, the CIMS Conversion Builder GUI provides a much simpler way to create a conversion definition. CIMS Conversion Builder also provides data validation features that ensure the conversion definition can be processed successfully by CIMS Conversion Engine. This section describes how to create a conversion definition using CIMS Conversion Builder.

To start CIMS Conversion Builder, click **Start ▸ Programs ▸ CIMS Server ▸ Conversion Builder** (if you installed CIMS Data Collectors in the default location). Or, from the CIMS Server Administrator main window, click **Chargeback Administration ▸ Processing ▸ Conversion Builder**.

Creating a Definition file

CIMS Conversion Builder provides the following tabs that walk you through each of the required and optional steps for creating a conversion definition:

- **Input tab.** Defines the input usage metering file data.
- **Output tab.** Defines the output file. If you are running CIMS Conversion Builder from the Universal collector, the output is always a CSR file.
- **Fields tab.** Defines the fields in the usage metering file.
- **Identifiers tab.** Defines the usage metering file fields to be used as identifiers in the output file.
- **Resources tab.** Defines the usage metering file fields to be used as resources in the output file.
- **Date/Time tab.** Defines the start and end date and time that appear in the output file.
- **Shifts tab.** Defines whether shift processing is enabled.

The following sections provide descriptions for each of the options on these tabs. For each tab option, the corresponding conversion definition statement is also provided.

Note • CIMS Lab provides a sample conversion definition (`SodaLogDef.txt`) in `Processes\Universal` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8). See page 12-30 for examples of these tabs as they appear for `SodaLogDef.txt`.

Input Tab

Use the **Input** tab to enter the parameters for the usage metering file as shown in the following table. All parameters are required unless noted otherwise.

For an example of a configured **Input** tab, see page 12-30.

CIMS Conversion Builder Option	Definition File Statement	Description
Description	`Description=<description of this definition file>`	Briefly describes the purpose of the conversion definition. The field size is approximately 100 bytes. A description is optional and has no impact on the conversion process.
Input Type ■ Delimited ASCII Text File (default) ■ Fixed-length ASCII Text File ■ ODBC Query ■ Microsoft Access Database Query	`ProcessType=` ■ `DELIMITED` ■ `FIXED` ■ `ODBCQUERY` ■ `MSACCESS`	The type of data to be processed: data in an ASCII text file or data extracted from a database query. For ASCII files, delimited means the usage metering file record has fields separated by a delimiter, such as a Comma Separated Values (CSV) file. Fixed means the file record has fixed-length fields. The input type determines which of the following processing options appear. If the input type is an ASCII text file, continue to *ASCII Text File* on page 12-5. If the input type is a database query, skip to *Database Query* on page 12-7.

Table 12-1 • Input Tab

CIMS Conversion Builder Option	Definition File Statement	Description
ASCII Text File		
Input Filename	`InputFile=<path and file name of input file>`	This is the path and name of the input usage metering file. If you are running CIMS Conversion Engine from the Universal collector, the input path and file name in the conversion definition are not used. The input path and file name defined in the job script are used. **Important:** While the input path and file name are not required in the conversion definition if you are using the Universal collector, the statement `InputFile=` is still required in the conversion definition file. Make sure that you do not delete this statement from the file. If you are running CIMS Conversion Engine from Conversion Builder, enter the path and name for the input file. The maximum path that can be specified is approximately 250 bytes.
Record Delimiter ■ NEWLINE (default) ■ BLANKLINE ■ FORMFEED	`RecDelimiter =` ■ `NEWLINE` ■ `BLANKLINE` ■ `FORMFEED`	The character used to delimit records (normally NEWLINE). If fields are terminated by a new line, then set the record delimiter to BLANKLINE. You can select a delimiter from the list, or you can enter the ASCII keyboard character(s) for other delimiters. For example, `^I` for a tab.

Table 12-1 • Input Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Field Delimiter (does not appear for Fixed-length input type) ■ COMMA (default) ■ TAB ■ SEMICOLON ■ COLON ■ NEWLINE ■ SPACE	`Delimiter=` ■ `COMMA` ■ `TAB` ■ `SEMICOLON` ■ `COLON` ■ `NEWLINE` ■ `SPACE` ■ `<any character literal>`	The character used to delimit fields in a usage metering file. A field delimiter is required only for delimited files. You can select a delimiter from the list, or you can enter the ASCII keyboard character(s) for other delimiters. For example, a forward slash (/).
Text Field Qualifier (does not appear for Fixed-length input type) ■ DOUBLEQUOTE (default) ■ QUOTE ■ NONE	`TextQualifier=` ■ `DOUBLEQUOTE` ■ `QUOTE` ■ `NONE` ■ `<any character literal>`	The character used for fields with embedded delimiter characters. For example, if the field delimiter is COMMA and the field value is `"1,345"` then the text field qualifier is DOUBLEQUOTE. The quotation marks in this case mark the beginning and ending of the field value. Quote indicates a single quote qualifier. A text field qualifier is required only for delimited files. You can select a qualifier from the list, or you can enter the ASCII keyboard character(s) for other delimiters. For example, an asterisk (*).
Skip Initial Lines	`InitialSkipLineCnt=n`	The number of lines to skip before beginning to process a usage metering file. You can select a number from the drop-down list or type a number. This is useful in situations where there are a number of header lines preceding the actual data. The default is 0 (skip no lines).

Table 12-1 • Input Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Database Query		
ODBC Data Source (appears when ODBC Query input type is selected)	`InputFile=<name of ODBC data source>`	The ODBC database to be queried. The database must be listed in the Windows ODBC Data Source Administrator. Type the database name or click **Browse** to the select the database from the Select ODBC Data Source dialog box. If the database that you want to query is listed in the dialog box, click the database, and then click **OK**. If the database is not listed in the dialog box, click the **ODBC Data Source Administrator** button and add the database. For instructions, refer to *Appendix D, Adding Data Sources*.
Access Database (appears when Microsoft Access Database Query input type is selected)	`InputFile=<path and file name of database>`	The access database to be queried. The maximum path that can be specified is approximately 250 bytes.
ODBC User ID	`OdbcUid=<user ID>`	The user ID for the database (if required).
ODBC User Password	`OdbcPwd=<encrypted user password>`	The user password for the database (if required). The password is encrypted.
ODBC SQL Query	`OdbcQuery=<SQL query>`	The database query.

Table 12-1 • Input Tab (Continued)

Output Tab

Use the **Output** tab to enter the parameters for the output file as shown in the following table. All parameters are required unless noted otherwise.

For an example of a configured **Output** tab, see page 12-31

CIMS Conversion Builder Option	Definition File Statement	Description
Output Filename	`OutputFile=<path and file name of generated output file>`	This is the path and name of the output file. The output file must be stored in a process definition folder within the `Processes` folder. If you are running CIMS Conversion Engine from the Universal collector, the output path and file name in the conversion definition are not used. The job script defines the output path, and the output file name is determined by the `Scan.wsf` script. The output file name is either `CurrentCSR.txt` or `YYYYMMDD.txt`. See *Scan.wsf* on page 2-16. **Important:** While the output path and file name are not required in the conversion definition if you are using the Universal collector, the statement `OutputFile=` is still required in the conversion definition file. Make sure that you do not delete this statement from the file. If you are running CIMS Conversion Engine from Conversion Builder, enter the path and name for the output file. The maximum path that can be specified is approximately 250 bytes.

Table 12-2 • Output Tab

CIMS Conversion Builder Option	Definition File Statement	Description
Output Record Type ■ CIMS Server Resource Record (default) ■ CIMS Transaction Record	`OutRecType=` ■ `CBS` ■ `TRANS`	The CIMS record type in the output file. If you are running CIMS Conversion Engine from the Universal collector, you cannot create transaction records. However, you can create transaction records if you run CIMS Conversion Engine from CIMS Conversion Builder (see page 12-24). **Note:** Although CIMS Transaction records are supported, CIMS Lab recommends that you use the CIMS Server Resource (CSR) records. CSR records are more flexible and efficient than transaction records.
Resource Header (appears when CIMS Server Resource Record is selected)	`UnivHdr=<field name>`	The resource header defines the source of data. A resource header is not available in all usage metering files and is not required. Depending on whether the usage metering file contains a header, you can do the following: ■ If the records within the file *do not* contain a header, you can add a header here if you want a header to appear in the output file. Otherwise, leave this box blank. ■ If the records within the file *do* contain a header, you can select the header field from the drop down list (if the field is entered in the **Fields** tab), type the field name, or leave the field blank (if you do not want the header to appear in the output file).

Table 12-2 • Output Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Output standard server identifiers (appears when CIMS Server Resource Record is selected)	`WriteStandard ServerIdentifiers=` ■ `YES` ■ `NO`	When this check box is selected, the identifiers `SourceName` and `SourceLine` are added to the output file. `SourceName` shows the path for the source (for fixed-length, comma delimited or Access query input types) or the name of the data source (for the ODBC query type). `SourceLine` shows the line of the usage metering file that produced the record. Standard identifiers are optional. The values for these identifiers can be lengthy. If the length of the output records is a consideration, leave this check box clear (the default).
Audit Code Default (appears when CIMS Transaction Record is selected)	`AuditCodeDefault= <string literal>`	A string that is used to hold a default audit code value (see page 12-19). A default audit code is optional. The default audit code can be a maximum of eight characters and simply serves as a user-defined field that helps to identify the record (i.e., an employee code, service code, etc.). The audit code does not affect data processing in any way.

Table 12-2 • Output Tab (Continued)

Fields Tab

Use the **Fields** tab to define the fields in the usage metering file as shown in the following table.

The required parameters depend on the input type.

- For fixed-length usage metering files, the **Field Name**, **Starting Column** (starting position for the field), and **Length** parameters are required. The **Type** parameter is also required for date and time fields if you are using date and time fields as the start and/or date time in the output file records (see *Date/Time Tab* on page 12-21).
- For all other usage metering files, only the **Field Name** is required with the exception that the **Type** is also required for date and time fields if you are using the date and time fields as the start and/or date time in the output file records.

Note • If the input type is a database query, click Populate Field List Using Query. CIMS Conversion Builder automatically populates the fields.

For an example of a configured **Fields** tab, see page 12-32.

CIMS Conversion Builder Option	Definition File Statement	Description
Field Number	`Fieldn`	An incrementing sequence number used to uniquely identify a field.
Field Name	`<field name>`	The field name. The name must be a single word or abbreviation (for example, `ACCTCD` for account code).
Starting Column	`COL(n)`	The starting position for the field. The starting position number is required for fixed-length files. It is optional for delimited files.
Length	`LEN(n)`	The length of the field. The length is required for fixed-length files. It is optional for delimited files.
Implied Decimals	`DEC(n)`	The number of decimal digits for the field. For example, if the field value in the usage metering file is `10000` and the implied decimal count is 2, the resulting value in the output file is `100.00`.

Table 12-3 • Fields Tab

CIMS Conversion Builder Option	Definition File Statement	Description
Type (Date/Time)	`TYPE(date or time)`	Time and date fields require a `TYPE` declaration specifying the format of the time and date as they appear in the usage metering file. The type format used for time and date fields is dependent on whether the time and date are fixed length or variable length. **Time Fields** ■ **Fixed length.** A fixed length time format is one in which there are a fixed number of digits for the time. For example, 12:34, 01:15, etc. Fixed length time fields in a usage metering file *do not* require a separator character. However, if the field includes a separator character, for example, 12:34, you need to include the character in the type format. If the field does not include a separator, for example, 1234, the separator character is optional. You can use hour (H), minutes (M), and seconds (S) in the following format: HH, MM, SS (seconds are optional). The format must be preceded by "`T-`". Examples: `T-HHMMSS`, `T-HH:MM`

Table 12-3 • Fields Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Type (Date/Time) (continued)		**Time Fields (continued)** ■ **Variable length.** A variable length time format is one in which there *is not* a fixed number of digits for the time. For example, 12:34, 1:15 (no preceding 0), etc. Variable length time fields in a usage metering file require a separator character and the character must be included in the type format. You can use hour (H), minutes (M), and seconds (S) in the following format: H, M, and S (seconds are optional). The format must be preceded by "T-". Examples: T-H:M, T-H:M:S **Date Fields** ■ **Fixed length.** A fixed length date format is one in which there are a fixed number of digits for the date. For example, 12252002, 01012003, etc. Fixed length date fields in a usage metering file *do not* require a separator character. However, if the field includes a separator character, for example, 12/25/2002, you need to include the character in the type format. If the field does not include a separator, for example, 12252002, the separator character is optional. You can use any combination of year (Y), month (M), and day (D) in the following format: YY or YYYY, MM or MMM, and DD. The format must be preceded by a "D-". Examples: D-YYYYMMDD, D-MM/DD/YYYY

Table 12-3 • Fields Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
		Date Fields (continued) ■ **Variable length.** A variable length date format is one in which there *is not* a fixed number of digits for the date. For example, 12/25/2002, 1/01/2003 (no preceding 0), etc. Variable length date fields in a usage metering file require a separator character and the character must be included in the type format. You can use any combination of year (Y), month (M), and day (D) in the following format: Y, M, and D. The format must be preceded by a `"D-"`. Examples: `D-Y/M/D`, `D-M/D/Y` If a year contains only two digits, the century is determined by the following: ■ Years 0–29 are assumed to occur in the 2000s (2000–2029) ■ Years 30–99 are assumed to occur in the 1900s (1930–1999)

Table 12-3 • Fields Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Filter	`FILTERPATTERN(reg expression)`	A regular expression or a literal that the field must match, otherwise the record is rejected. Regular expressions are used frequently in some utilities and programming languages such as grep, sed, awk, and Perl. The regular expression `FILTERPATTERN` is a subset of full regular expressions available in other tools and can consist of the following metacharacters: ^–Matches to beginning of field $–Matches to end of field *–Matches zero or more occurrences of the preceding literal .–Matches any character !–If this is the first character in an expression, it negates the outcome of the regular expression. That is, the expression is not matched. **Example** A usage metering file contains records with one of two account codes: 01100 or 01200. If you want just those records that contain the account code 01200, you could use the regular expressions ^012, 200$, 01*200, 0.2, !01100 (among others) or the literal 01200.

Table 12-3 • Fields Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Parse—Character	`PARSECHAR (character)`	The character used to split a string in the field. For example, to split a URL, enter the / character for this parameter. The character delimits the end of a word. **Parse—Character** and **Parse—Word Number** work together to parse a "word" from a string. For example, if you want only `cimsnt.asp` in the following example: `http://www.cimslab.com/cimsnt.asp` The parse character is / and the parse word is the fourth word in the string as follows: Word 1=`http:` Word 2=`null` Word 3=`www.cimslab.com` Word 4=`cimsnt.asp` In this case, you enter a 4 in the **Parse—Word Number** box.
Parse—Word Number	`PARSEWORD(n)`	The number of the word in the string that should be split by the parse character (see the preceding example for **Parse—Character**) and returned as the field value. The character delimits the end of a word. If the value for the parse word number is greater than the number of words indicated by the parse character, the last word in the string is returned. For example, if you entered a parse word number of `5` for the preceding example, the field value would be `cimsnt.asp` (there is no fifth word).

Table 12-3 • Fields Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
Insert Field		Inserts a field above the selected field.
Remove Field		Removes the selected field.
Populate Field List Using Query (appears when a database query input type is selected)		Automatically populates the field list with fields from the database. You can then change the field names if needed.

Table 12-3 • Fields Tab (Continued)

Identifiers Tab

Use the **Identifiers** tab to define the fields that are identifiers. Identifier fields are used as literals or lookup keys in the account code conversion in CIMS Server.

For an example of a configured **Identifiers** tab, see page 12-33.

CIMS Conversion Builder Option	Definition File Statement	Description
Field Number	`IDFIELDn`	An incrementing sequence number used to uniquely identify an identifier.
Field Name	`<field name>`	The field used as an identifier. To select the fields that you want to use as identifiers, click the drop-down arrow in the **Field Name** box. (The drop-down arrow appears when you click the box.)
Insert Field		Inserts a field above the selected field.
Remove Field		Removes the selected field.

Table 12-4 • Identifiers Tab

Resources Tab

Use the **Resources** tab to define the fields that represent resource usage. For example, a field that represents CPU time, transactions processed, or lines printed.

For an example of a configured **Resources** tab, see page 12-34.

CIMS Conversion Builder Option	Definition File Statement	Description
Field Number	`RSFIELDn`	An incrementing sequence number used to uniquely identify a resource.
Field Name	`<field name>`	The field containing a resource. To select the fields that you want to use as resources, click the drop-down arrow in the **Field Name** box. (The drop-down arrow appears when you click the box.)
Rate Code	`RATECODE(code)`	The rate code represents the resource units being reported by the field. To enter a rate code for the field, click the **Rate Code** box. Click the (...) button and do one of the following: ■ If a field within the usage metering file contains the rate code, click the field name. ■ If the rate code is contained in the CIMS Rate table of the CIMS Server database (see *ODBC Data Source Name* on page 12-20), click the existing rate code. ■ If the rate code is not contained in the usage metering file or the database, type the rate code name in the lower box. Do not use the same name for both the resource field and the rate code. **Important:** If you select a rate code from the usage metering file or create a new rate code, *you must add the rate code to the CIMS Rate table.* Rate codes that do not appear in the CIMS Rate table are not included in CIMS Server invoices and other reports.

Table 12-5 • Resources Tab

CIMS Conversion Builder Option	Definition File Statement	Description
Audit Code (appears when CIMS Transaction Record is selected as the output record type)	`AUDITCODE(code)`	A literal value specifying the audit code used to track this resource value. An audit code is optional. The default audit code can be a maximum of eight characters and simply serves as a user-defined field that helps to identify the record (i.e., an employee code, service code, etc.). The audit code does not affect data processing in any way. To enter a audit code for the field, click the **Audit Code** box. Click the **(...)** button and do one of the following: ■ If a field within the usage metering file contains the audit code, click the field name. ■ If you want to use the default audit code entered in the **Output** tab, click DEFAULT. ■ If you want to enter an audit code, type the code in the lower box.
Insert Field		Inserts a field above the selected field.
Remove Field		Removes the selected field.

Table 12-5 • Resources Tab (Continued)

CIMS Conversion Builder Option	Definition File Statement	Description
ODBC Data Source Name	`RateOdbcDsn=<name of database>`	The default ODBC database for the CIMS Rate table is `CIMSServer`. If you want to use the CIMS Rate table from another database, do one of the following: ■ Type the database name (the database must be listed in the Windows ODBC Data Source Administrator). ■ Click **Browse** to the select the database from the Select ODBC Data Source dialog box. If the database that you want to use is listed in the dialog box, click the database, and then click **OK**. If the database is not listed in the dialog box, click the **ODBC Data Source Administrator** button and add the database. For instructions, refer to *Appendix D, Adding Data Sources*.
ODBC User ID	`OdbcUid=<user ID>`	The user ID for the database (if required).
ODBC Password	`OdbcPwd=<encrypted user password>`	The user password for the database (if required). The password is encrypted.

Table 12-5 • Resources Tab (Continued)

Date/Time Tab

Use the **Date/Time** tab to define the start and end date and time that appear in the output file records.

For an example of a configured **Date/Time** tab, see page 12-35.

CIMS Conversion Builder Option	Definition File Statement	Description
Record Date Low and High	RecDateLo= ■ SYSTEM ■ <field name> ■ RNDATE ■ CURDAY ■ CURWEK ■ CURMON ■ PREDAY ■ PREWEK ■ PREMON RecDateHi= ■ SYSTEM ■ <field name>	Determines the start and end date that appear in the output file records. You can select one of the following: ■ System Date. The computer system date is used. This is the default. ■ One of the following date keywords. If you select a keyword in the **Record Date Low** box, you cannot select values in the **Record Date High** box or the **Record Time** boxes. These boxes are unavailable. • **Run Date (Today).** The start and end date is the current day. • **Previous Day to Current Day.** The start date is the previous day and the end date is the current day. • **Current Week/Month.** The start date is the first day of the current week/month and the end date is the last day of the current week/month. • **Previous Day.** The start and end date are the previous day. • **Previous Week/Month.** The start date is the first day of the previous week/month and the end date is the last day of the previous week/month. ■ A date field (if defined in the **Fields** tab). The value in the date field is used as the date.

Table 12-6 • Date/Time Tab

CIMS Conversion Builder Option	Definition File Statement	Description

Record Time Low and High	`RecTimeLo=`
	■ `SYSTEM`
	■ `<field name>`
	■ `ENTIRE`
	`RecTimeHi=`
	■ `SYSTEM`
	■ `<field name>`
	■ `ENTIRE`
		Determines the start and end time that appear in the output file records. Note that if a keyword is selected in the **Record Date Low** box, the **Record Time** boxes are unavailable.
		You can select one of the following:
		■ System Time. The computer system time is used. This is the default.
		■ A time field (if defined in the **Fields** tab). The value in the time field is used as the time.
		■ Entire Day. Defines the start time as `00:00:00` and the end time as `23:59:59`.

Table 12-6 • Date/Time Tab (Continued)

Shifts Tab

Use the **Shift** tab to define whether shift processing is enabled. In shift processing, a shift character is entered in the Shift Code field (for CSR records) or appended to the existing rate code (for CIMS Transaction records). Using shifts enables you to charge different rates for different work shifts.

When entering shifts:

- You may enter a maximum of 5 shifts per day.
- The shift characters can be any valid character, and the times must be listed in 4-character, 24-hour format.

For an example of a configured **Shifts** tab, see page 12-36.

CIMS Conversion Builder Option	Definition File Statement	Description
Shift Processing Enabled	`ShiftsEnabled=` ■ `YES` ■ `NO`	If the check box is selected, the use of shifts is enabled. If the check box is clear, the use of shifts is not enabled.
Shift Field	`ShiftField=<field>`	The name of a time field which is used to generate a shift character in the output file. If a time field is not specified, the output file record start time is used to generate the shift character (see *Date/Time Tab* on page 12-21).
Shift Char	`Shift<day>= DEFINE <shift char> <end time> [ <shift char> <end time> ...]`	The number (1–9) that represents the shift, for example, 1 for the first shift, 2 for the second shift, etc.
End Time		The time that the shift ends.

Table 12-7 • Shifts Tab

Opening a Conversion Definition

To open a conversion definition, click **File ▸ Open Conversion Definition**.

Saving a Conversion Definition

To save a new conversion definition, click **File ▸ Save As**. To save changes to an existing definition, click **File ▸ Save**.

Viewing Conversion Definitions

You can view the conversion definition, usage metering file, and output file for the current definition directly from CIMS Conversion Builder. To view a file, click **File ▸ View *File Type***.

Running CIMS Conversion Engine

Once you have created a conversion definition for a usage metering file, you can run CIMS Conversion Engine directly from CIMS Conversion Builder to ensure that the output file contains the data that you want.

To run CIMS Conversion Engine from CIMS Conversion Builder, click **File ▸ Run Conversion**. The output file is created and placed in the location specified on the **Output** tab (see *Output Tab* on page 12-8).

Setting Up and Running the Universal Collector

This section provides the information you need to set up and run the Universal collector.

Adding Resource Rate Codes to the CIMS Rate Table

Because the resources collected by CIMS Data Collector are user-defined in the conversion definition and not pre-defined by CIMS Lab, you need to add the rate codes for the resources to the CIMS Rate table.

Rate codes that do not appear in the CIMS Rate table are not included in CIMS invoices and other reports. You cannot load an output file into the CIMS Server database until at least one rate code from the file is added to the CIMS Rate table.

To add rate codes, in the CIMS Server Administrator main window, click **Chargeback Administration ▸ Chargeback Table Maintenance ▸ Rate Codes** and follow the instructions in the *CIMS Server Administrator's Guide*.

Setting Up the Universal Collector

Each conversion definition that you create must be stored in a separate process definition folder with a separate job script.

If you have a single or multiple usage metering files and you are using the same conversion definition for all the files, you can use the default `Universal` process definition folder and modify the `JobUniversal.wsf` script as needed. The `Universal` process definition folder and `JobUniversal.wsf` script are in the `Processes` folder (see *About the Processes Folder* on page 2-8).

If you have multiple usage metering files requiring different conversion definitions, you need to create a separate process definition folder and job script for each conversion definition. To create the process definition folders and job scripts, simply copy and rename the `Universal` process definition folder and rename and modify the `JobUniversal.wsf` script for each definition. The job script name should begin with `Job` followed by the name of the process definition folder. For example, if the folder name is `ABCSOFT`, the name of the job script should be `JobABCSOFT.wsf`.

There are no changes required for the collector script, `Universal.wsf`. If you installed CIMS Data Collectors in the default location, the `Universal.wsf` script is in `C:\Program Files\CIMSLab\Collectors\Universal`.

Setting Up the JobUniversal.wsf Script

Note • The information in this section is applicable whether you are using the `JobUniversal.wsf` script or a copy of the script.

The `JobUniversal.wsf` script calls and passes parameters to the `Universal.wsf`, `Scan.wsf`, `ProcCIMS.wsf`, and `CleanUp.wsf` scripts.

The parameters required for the `Universal.wsf` script are described in the following section. The parameters required for the remaining scripts are described in the *Scripts Architecture* section beginning on page 2-13.

Setting the Parameters for the Universal.wsf Script

The `Universal.wsf` script requires the parameters shown in the following table.

Parameter	Description/Values
`LogDate`	Because the input log file is specified by the job script or the conversion definition, this parameter is not used to select log files. However, this parameter determines the date that appears in the file name of the CSR file that is processed by the `Scan.wsf` script. For example, if the value is `PREDAY`, the previous day's date appears in the file name. The values are: ■ `preday` (previous day) ■ `premon` (previous month) ■ `rndate` (current day) ■ `curday` (current day and previous day) ■ `curmon` (current month) ■ date in `yyyymmdd` format This parameter is entered in the command line when running the collector.
`RetentionFlag`	This parameter is for future use.
`Feed`	The name of the server that contains the log files that you want to process or `SELF` if the log files are on the same server as the `Universal.wsf` script. A subfolder with the same name as the server is automatically created in the process definition folder (see the `OutputFolder` parameter). This subfolder is used to store the CSR files that are created from the log files (see *Feed Subfolder* on page 2-12).

Table 12-8 • Universal Script Parameters

Parameter	Description/Values
`OutputFolder`	The process definition folder for the collector. This is the location of the final CSR file that is created by the `Scan.wsf` script (see page 2-16). For more information about the process definition folder, see *Processes Architecture*.
`ConvEngDefName`	The location of the conversion definition file.
`InputFileName`	The conversion definition specifies the path and name of the input usage metering file that is processed by CIMS Conversion Engine.

Table 12-8 • Universal Script Parameters (Continued)

Running the Universal Collector

To run CIMS Universal Data Collector, use the `Nightly.bat` program (this is shipped as `SampleNightly.bat`, see *Nightly.bat and Monthly.bat* on page 2-13). You can run this program directly from the command prompt or you can use Windows Scheduled Tasks to schedule the program to run automatically (see *Appendix C, Running Batch Scripts* for instructions).

Make sure that the collector files are set up correctly as described in the *Setting Up the System (System Architecture)* section beginning on page 2-3. The name of the process definition folder for the Universal collector must be included in the list of jobs in the `Nightly.wsf` script (this is shipped as `SampleNightly.wsf`, see *Nightly.wsf and Monthly.wsf* on page 2-15) and must be uncommented as shown:

```
DoJob("Universal")
```

Example Files

An example usage metering file, conversion definition, and output CSR file are in `Processes\Universal` where the folder `Processes` can be in any location (see *About the Processes Folder* on page 2-8). These files are named `SodaLog.txt`, `SodaLogDef.txt`, and `CurrentCSR.txt`, respectively.

The following sections describe each of these files.

Log File—SodaLog.txt

The file `SodaLog.txt` is a log file for the fictional "ACME Soda Tracker" program. This program monitors the refrigerator in the break room and generates a log entry every time someone removes a soda can. Each entry records the date, time, name of the person removing the soda, and the number of soda cans removed. The log file contains the following data:

```
01062004        05:27        MARY         1
01062004        07:13        RON          1
01062004        10:20        BERT         1
01062004        11:01        JANICE       1
01062004        12:23        JANICE       1
01062004        12:34        RANDY        1
01062004        16:02        TONY         1
01062004        17:37        JERRY        1
```

Conversion Definition File–SodaLogDef.txt

The following sections describe the `SodaLogDef.txt` as viewed in CIMS Conversion Builder and Notepad.

Conversion Definition Viewed in CIMS Conversion Builder

The information contained in the `SodaLogDef.txt` file is grouped by tabs in CIMS Conversion Builder as shown in this section. Note that the examples in this section reflect the options set in the `SodaLogDef.txt` file and that not all of the options available on the tabs are described. For a detailed description of each tab option, see *Creating a Conversion Definition Using CIMS Conversion Builder* on page 12-3.

Input Tab

The **Input** tab defines the description (optional), file type, and path for the `SodaLog.txt` file.

New records in the `SodaLog.txt` file are indicated by a newline character, the field delimiter is a tab, and no text field qualifier is needed.

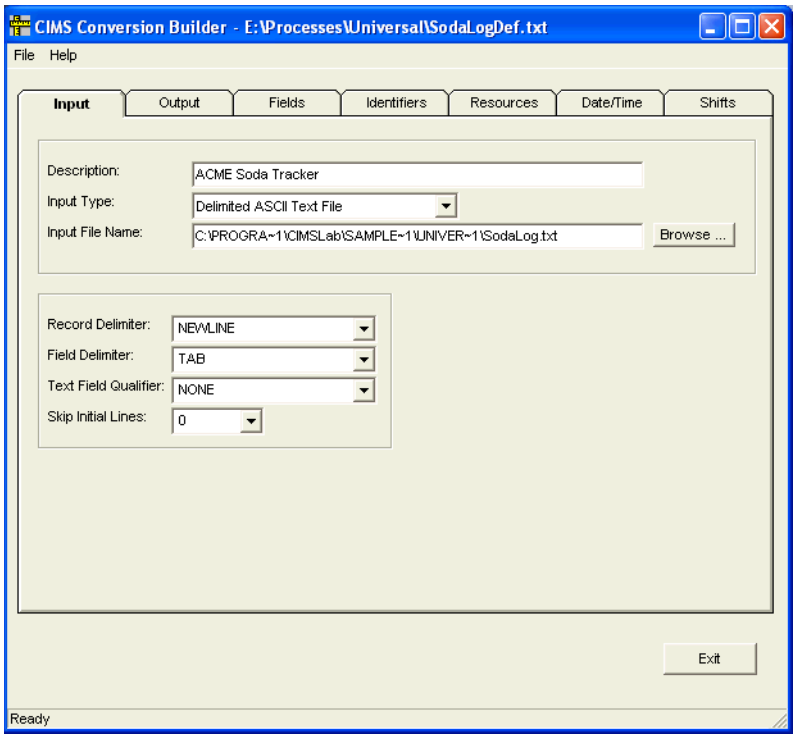

Figure 12-1 • Input Tab

Output Tab

A CSR file named `CurrentCSR.txt` will be generated and stored in the `Universal` folder. The header `SAMPLE` will appear in the CSR records.

Note • The following example shows the output file name as defined in the default `SodaLogDef.txt` conversion definition that is shipped by CIMS Lab. If use the Universal collector to convert the sample `SodaLog.txt` file, the collector will automatically update the output file path as defined in the `JobUniversal.wsf` script.

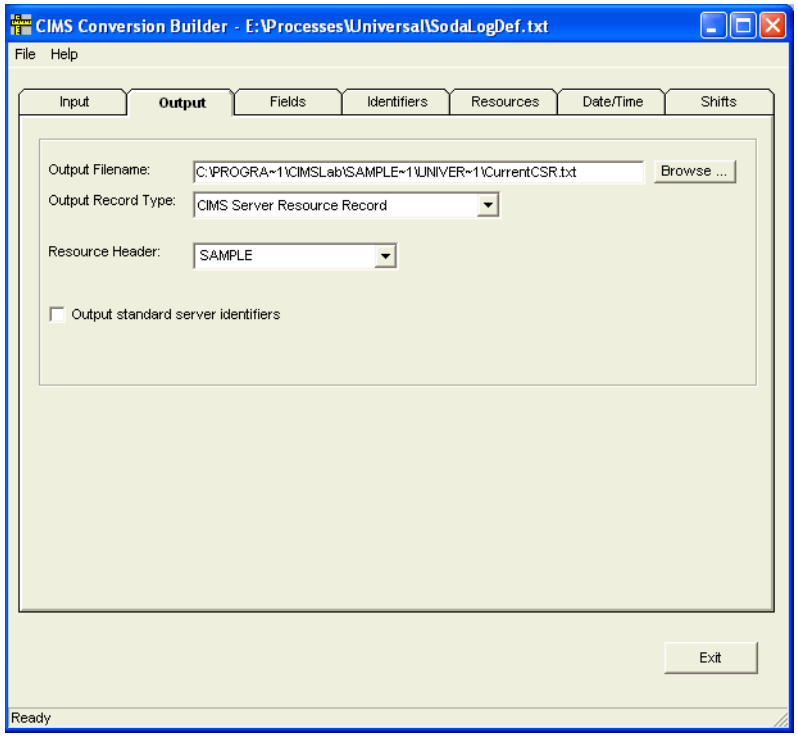

Figure 12-2 • Output Tab

Fields Tab

This log has four fields (the date, time, user name, and number of sodas removed). The field names DATE, TIME, USER, and SODA are assigned to the fields respectively. Note that the formats for the DATE and TIME fields are declared in the **Type** field (see page 12-12).

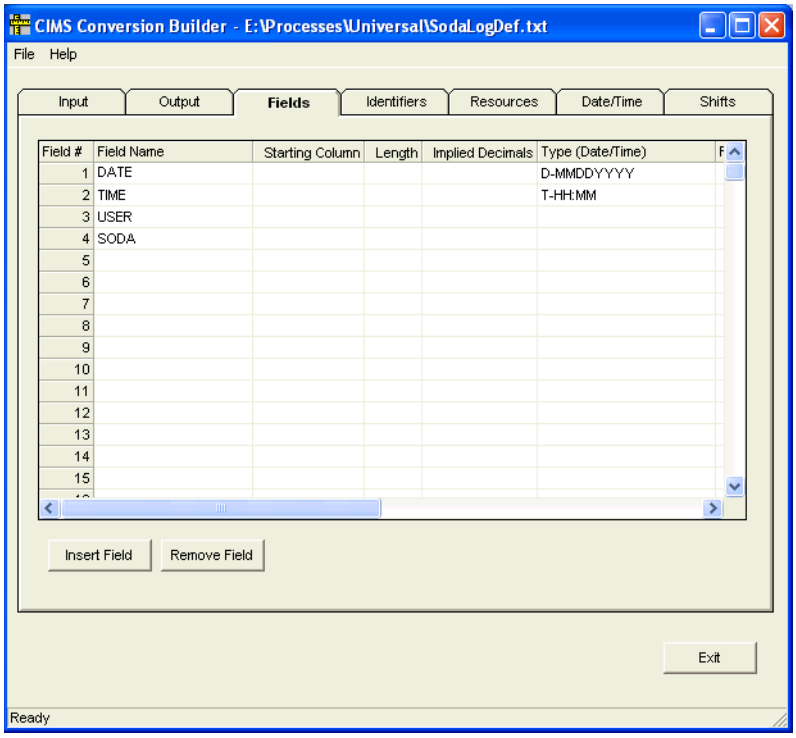

Figure 12-3 • Fields Tab

Identifiers Tab

The identifiers in the log file are contained in the DATE, TIME, and USER fields.

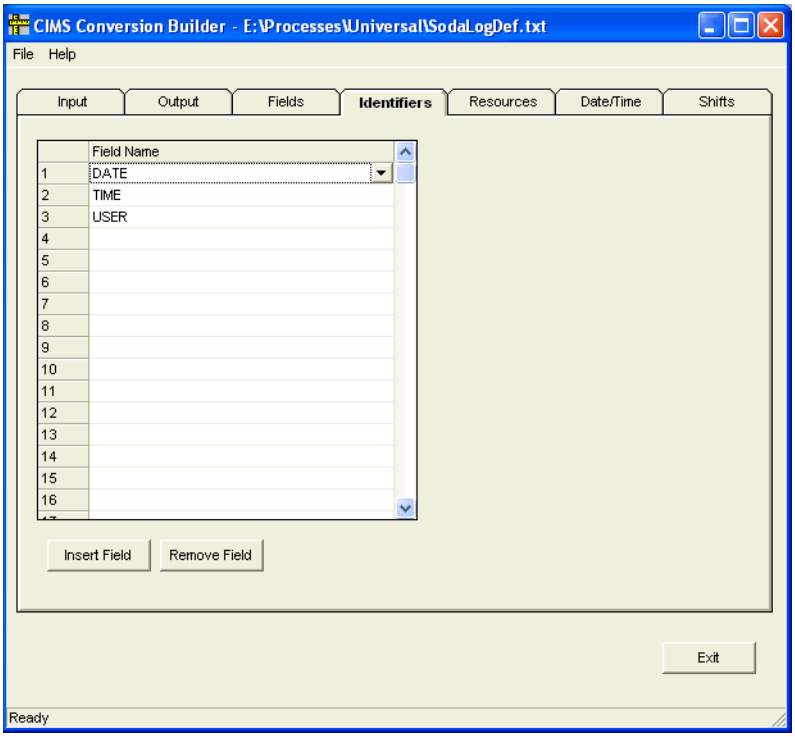

Figure 12-4 • Identifiers Tab

Resources Tab

The field SODA represents the resources being consumed. A rate code named EMPBEV (for employee beverage) has been assigned to identify the resource. This rate code appears in the invoices and other reports generated by CIMS Server.

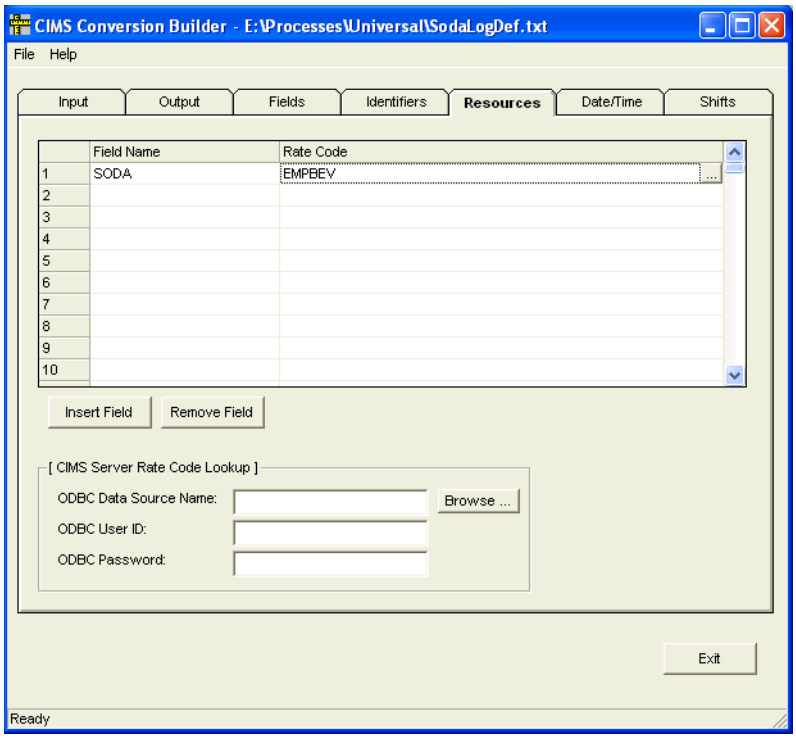

Figure 12-5 • Resources Tab

Date/Time Tab

The system date will appear as the start/end date and time in the CSR records.

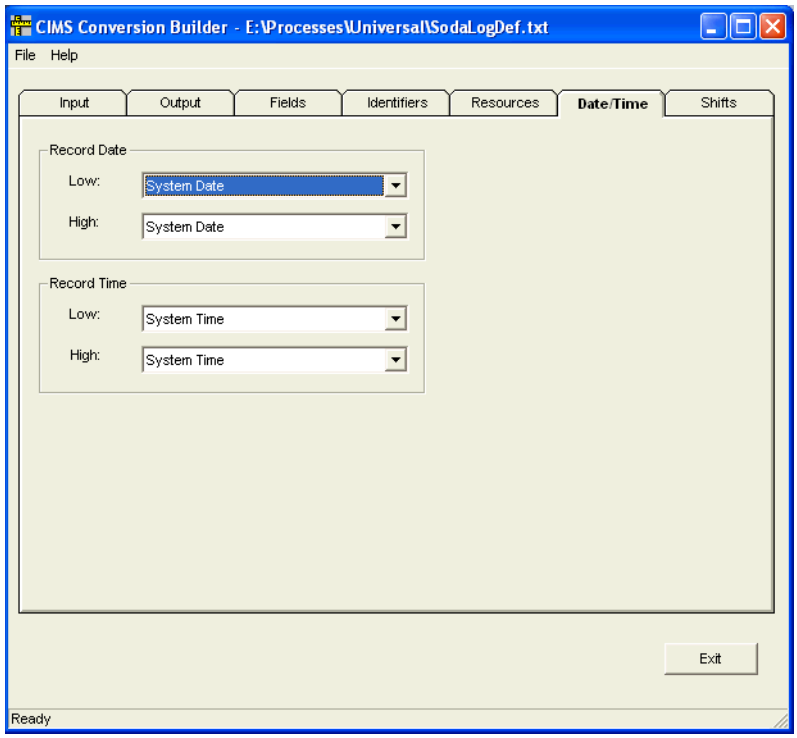

Figure 12-6 • Date/Time Tab

Shifts Tab (Optional)

Rate shifts allow you to set different rates based on the time of day. For example, if employees are charged for sodas, the rate might differ depending on the shift. In this example, the TIME field is entered in the **Shift Field** box. This specifies that shift character that appears in the output file records is determined by the time in the TIME field. If **None** is selected in the **Shift Field** box, the shift character is determined by the start date in the output file record (see *Date/Time Tab* on page 12-21).

Shifts are represented by a numeric value 1–9. This example indicates that on all records for Monday through Friday, shift 3 is from midnight to 8 a.m., shift 1 is from 8 a.m. to 4 p.m., and shift 2 is from 4 p.m. to midnight.

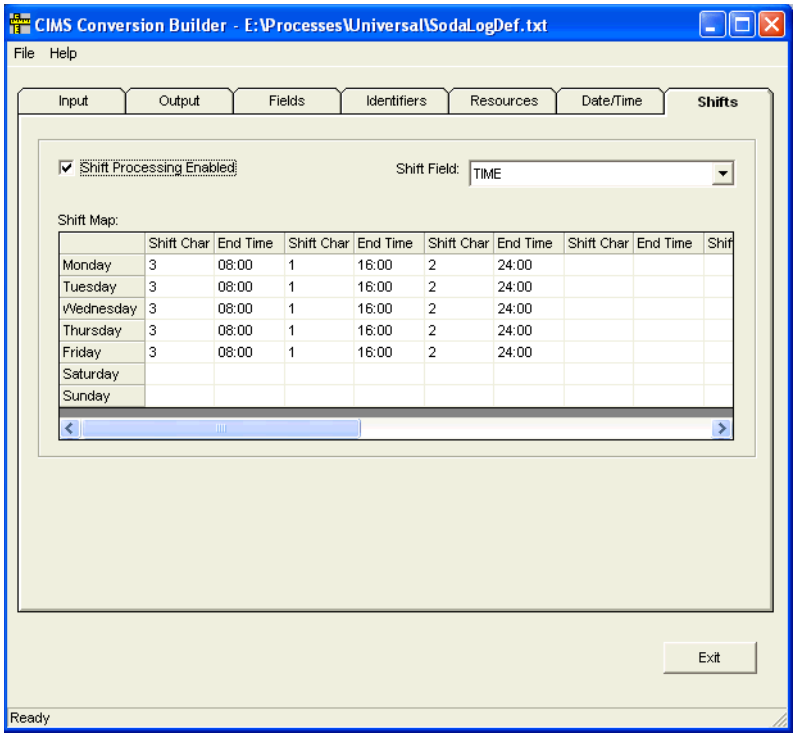

Figure 12-7 • Shifts Tab

Conversion Definition Viewed in Notepad

The file SodaLogDef.txt contains ASCII text in the same format as a Windows .INI file. Each line in the file holds a single statement and must end with the CRLF pair.

```
[Control]
Description=ACME Soda Tracker
InputFile=C:\PROGRA~1\CIMSLab\SAMPLE~1\UNIVER~1\SodaLog.txt
OutputFile=C:\PROGRA~1\CIMSLab\SAMPLE~1\UNIVER~1\CurrentCSR.txt
OutRecType=CBS
ProcessType=DELIMITED
Delimiter=TAB
RecDelimiter=NEWLINE
InitialSkipLineCnt=0
TextQualifier=NONE
RecDateLo=SYSTEM
ShiftField=TIME
ShiftMON=DEFINE 3 0800 1 1600 2 2400
ShiftTUE=DEFINE 3 0800 1 1600 2 2400
ShiftWED=DEFINE 3 0800 1 1600 2 2400
ShiftTHU=DEFINE 3 0800 1 1600 2 2400
ShiftFRI=DEFINE 3 0800 1 1600 2 2400
ShiftsEnabled=YES
UnivHdr=SAMPLE
WriteStandardServerIdentifiers=NO

[Layout]
Field1=DATE TYPE(D-MMDDYYYY)
Field2=TIME TYPE(T-HH:MM)
Field3=USER
Field4=SODA
IDField1=DATE
IDField2=TIME
IDField3=USER
RSField1=SODA RATECODE(EMPBEV)
```

The conversion definition is divided into two sections: [CONTROL] and [LAYOUT]. The [CONTROL] section includes option statements that guide the processing performed by CIMS Conversion Engine. The [LAYOUT] section describes the data fields within the log file. For a description of the statements and values used in the conversion definition, see *Creating a Conversion Definition Using CIMS Conversion Builder* on page 12-3.

Comments may be added on any line in the conversion definition. The line must start with a semicolon (;) in column 1. For example:

```
; This is a comment line
```

Output File—CurrentCSR.txt

If you ran the Universal collector on April 19, 2004, the output CSR file, `CurrentCSR.txt`, created from the `SodaLog.txt` log would contain `20040419` in the start and end date fields and the system time in the start and end time fields as shown in the following example:

```
SAMPLE,20040419,20040419,10:52:27,10:52:27,3,3,DATE,"01062004",TIME,"05:27",USER,"MARY",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,3,3,DATE,"01062004",TIME,"07:13",USER,"RON",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,1,3,DATE,"01062004",TIME,"10:20",USER,"BERT",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,1,3,DATE,"01062004",TIME,"11:01",USER,"JANICE",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,1,3,DATE,"01062004",TIME,"12:23",USER,"JANICE",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,1,3,DATE,"01062004",TIME,"12:34",USER,"RANDY",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,2,3,DATE,"01062004",TIME,"16:02",USER,"TONY",1,EMPBEV,1
SAMPLE,20040419,20040419,10:52:27,10:52:27,2,3,DATE,"01062004",TIME,"17:37",USER,"JERRY",1,EMPBEV,1
```

13

Contacting Technical Support

The CIMS Lab Technical Support department is here to answer your questions on any aspect of CIMS Lab products.

CIMS Lab technical support can be reached in the following ways:

- **Telephone**: (800) 283-4267 in USA and Canada; 916-783-8525 International
- **Email**: support@cimslab.com
- **Fax request**: (916) 783-2090

International customers may contact one of our authorized international partners. Contact CIMS Lab for more information.

In addition, customers may visit the Customer Area on our Web site for product downloads, updates, technical documentation, and password information. We are on the Web at http://www.cimslab.com.

A

CIMS Aggregation Engine API

About CIMS Aggregation Engine

CIMS Aggregation Engine (`CIMSAggregation.dll`) is a COM object that aggregates the records in the log file by identifier values. CIMS Aggregation Engine provides methods for uniquely identifying an aggregate within a log file, summarizing and storing information about the aggregate, and writing the aggregate information to a CIMS Server Resource (CSR) file.

Aggregation reduces the amount of data from a log file that must be processed by CIMS Processing Engine, thus reducing processing time. This is especially beneficial for log files that are created daily and contain gigabytes of data.

CIMS Aggregation Engine is designed to be called by compiled code or scripts that pass lists of identifier names, identifier values, rate codes, and resource values from the log file, as well as optional start and end dates. CIMS Aggregation Engine then generates an aggregation key for each unique set of matching identifier values. For example, three aggregation keys, `BERTACME1`, `JANICEACME1`, and `RANDYACME1` would be generated for the following log file records. The log file is generated by a fictional software program, "ACME Soda Tracker".

```
ACMESODA,20021031,20021031,11:02:43,,2,User,BERT,Machine,ACME1,1,SODA,1

ACMESODA,20021031,20021031,11:02:57,,2,User,JANICE,Machine,ACME1,1,SODA,1

ACMESODA,20021031,20021031,12:05:34,,2,User,JANICE,Machine,ACME1,1,SODA,1

ACMESODA,20021031,20021031,12:10:05,,2,User,RANDY,Machine,ACME1,1,SODA,1
```

Once an aggregation key is created, resource values passed to CIMS Aggregation Engine are matched to the key and added to the existing aggregated resource values associated with the key. For example, in the preceding log file, the second and third records share the same aggregation key. CIMS Aggregation Engine would aggregate these records to produce a resource value of `2` for the rate code `SODA`.

After all log file records have been passed to CIMS Aggregation Engine, the engine writes a CSR file.

About the Aggregation Algorithm

The base aggregation algorithm used by CIMS Aggregation Engine is the Repeated Scanning algorithm[1]. The algorithm maintains as many aggregates in memory as possible. When no more aggregates can fit into memory, new aggregates are written to a work file. Only relevant information from each record, such as identifier and resource values, are written to the work file.

[1] See *Grouping and Duplication Elimination: Benefits of Early Aggregation*, Microsoft Corporation, January 1997, http://www.research.microsoft.com/~palarson.

Processing the Log File

CIMS Aggregation Engine continues to read the log file until it reaches the end of the file. Existing or new aggregates found in the log file are updated in main memory. When CIMS Aggregation Engine reaches the end of the file, the aggregates stored in memory are written out to the CSR file and cleared from memory. If a work file was written, a loop is entered to process the work file repeatedly until it is no longer required (see *Processing the Work File*).

Processing the Work File

The base aggregation algorithm is also used to process the work file with the exception that CIMS Aggregation Engine handles all Input/Output (I/O). The number of passes required to process the work file is the total number of aggregate entries in the input log file divided by the number of aggregate entries that will fit in memory. It is expected that the number of aggregate entries will be low compared to the number of records in the input log file.

If the number of work file passes is high, the speed of aggregation is reduced because each generation and subsequent processing of the work file results in additional I/O, which is slower than main memory. There is an extension to the algorithm that hash partitions the single work file into multiple work files. By applying a hash function to the aggregation key, records belonging to the same aggregate are grouped together in a separate work files. The work files are then processed based on size, smallest file first. It is assumed that smaller work files will generate fewer future work files, thereby reducing overall work file data to be processed.

CIMSAggregation Interfaces

`CIMSAggregation` uses the following interfaces:

- `TypedEngine`. This strongly-typed interface is used by programming languages that support strong types.
- `ScriptingEngine`. This weakly-typed (variant) interface is used primarily by scripting applications as scripting is based on a weakly-typed system. This interface delegates its calls to an instance of `TypedEngine`.
- `ExceptionFile`. This interface produces exception files containing unprocessed records.

The properties and methods for each interface are described in the following sections.

TypedEngine and ScriptingEngine Interfaces

Except where noted, the properties and methods described in this section are contained in both the `TypedEngine` and `ScriptingEngine` interfaces. However, in the `ScriptingEngine` interface, all types are passed and returned as variants.

About Specifying Dates and Times

When using the `TypedEngine` or `ScriptingEngine` interface, there are three ways to specify the dates and times that appear in the CSR file:

- The `DateKeyword` property. This property overrides the `DateStart` and `EndDate` properties and the `AddEntry` or `AddEntries` method date parameters. For this property, the start time is 00:00:00 and the end time is 23:59:59.
- The `DateStart` and `DateEnd` properties. If there is no `DateKeyword` property, the `DateStart` and `DateEnd` properties override the date parameters of the `AddEntry` or `AddEntries` method. For these properties, a time can be specified as part of the date.
- The date parameters specified by the `AddEntry` or `AddEntries` method. If there is no `DateKeyword` property or `DateStart` and `DateEnd` properties, the these methods are used. For these parameters, a time can be specified as part of the date.

If the none of the preceding properties or parameters are specified, the start time is 00:00:00 and the end time is 23:59:59.

TypedEngine and ScriptingEngine Interface Properties

AggregationList

Returns aggregated records in an array rather than writing them to a CSR file. The aggregate data can then be modified if needed.

Note that CIMS Aggregation engine does not process modified records. To write modified records to a CSR file, use a script. CIMS Lab provides the class `CSRWriter` in the `CIMSLib.wsf` script to write CSR files.

Syntax

object`.AggregationList`

Parameters

None.

Comments

To use this property, the aggregates must fit in memory.

Example

Retrieve the array:

```
Dim List
List = AggregationEngineObject.AggregationList
```

The two-dimensional array is returned in the same order as the CIMS Server Resource Record. For example, entry 0 in the array might appear as follows:

```
AggList (0,  0) = DEMOARRY                (Resource Header [String])
AggList (0,  1) = 4/1/2003                (Start Date/Time [Date])
AggList (0,  2) = 4/30/2003 11:59:59 PM   (End Date/Time [Date])
AggList (0,  3) = 3                       (Number of Identifiers [Long])
AggList (0,  4) = ServerId                (Identifier Name [String])
AggList (0,  5) = Server#000003           (Identifier Value [String])
AggList (0,  6) = UserId                  (Identifier Name [String])
AggList (0,  7) = User#000003             (Identifier Value [String])
AggList (0,  8) = FileNumber              (Identifier Name [String])
AggList (0,  9) = FileNum1                (Identifier Value [String])
AggList (0, 10) = 2                       (Number of Resources [Long])
AggList (0, 11) = RES1                    (Rate Code [String])
AggList (0, 12) = 6                       (Resource Value [String])
AggList (0, 13) = RES2                    (Rate Code [String])
AggList (0, 14) = 0.9                     (Resource Value [String])
```

DataValidation

Returns or sets a Boolean value that indicates whether the incoming data should be verified. Verification includes scanning all input for invalid character data.

Syntax

object.DataValidation [=*value*]

Parameter

Value

A Boolean value that indicates whether incoming data should be checked.

Comments

The default value is False, data should not be verified. Verifying incoming data may slow down the aggregation process.

If the value is set to True:

- The TypedEngine interface makes the following checks:
  - The number of identifiers passed to the AddEntry or AddEntries method must match the number of identifiers declared by the DefineIdentifier method.
  - The number of resources passed to the AddEntry or AddEntries method must match the number of resources declared by DefineResource method.
- The ScriptingEngine interface makes the following checks:
  - StartDate and EndDate parameters passed to the AddEntry or AddEntries method are checked to ensure that they are valid dates.
  - Resource values passed to the AddEntry or AddEntries method are checked to ensure that they are numeric values.

DateAggregation

Returns or sets a the date field to aggregate on.

Syntax

object.DateAggregation [=*value*]

Parameter

Value

A value specifying the date fields to aggregate on. Valid value is None, StartDate, EndDate, or Both. The enumeration values are:

- EDateAggregation_None = 1
- EDateAggregation_StartDate = 2
- EDateAggregation_EndDate = 3
- EDateAggregation_Both = 4

Comments

The default value is to aggregate on both date fields, EDateAggregation_Both.

If EDateAggregation_None or EDateAggregation_Both is specified, the CIMS Server Resource Record will contain a minimum of the start date value and the maximum of the end date value.

If EDateAggregation_StartDate is specified, the CIMS Server Resource Record start date/time fields will contain the minimum of the start date value. The CIMS Server Resource Record end date/time fields will contain the end date/time from the first record.

If EDateAggregation_EndDate is specified, the CIMS Server Resource Record end date/time fields will contain the maximum of the end date value. The resource record start date/time fields will contain the start/time date from the first record.

To perform date aggregation, the date parameters must be specified in the AddEntry or AddEntries method.

DateEnd

Returns or sets a default date value that specifies the ending date field to be written to the records in the CSR file.

Syntax

object.DateEnd [=*value*]

Parameter

Value

A date value specifying the date end value of the CIMS Server Resource Record.

Comments

This property is overridden if the DateKeyword property is specified.

DateKeyword

Returns or sets a string value that specifies a keyword that determines the date range to use for date field values to be written to the records in the CSR file.

Syntax

object.DateKeyword [=*value*]

Parameter

Value

A pre-defined keyword value. Valid values are:

- "**RNDATE" or "**CURDAY"—Sets date range based on the run date.
- "**CURDAY"—Sets date range based on the run date.
- "**CURWEK"—Sets date range based on the run week (Sun–Sat).
- "**CURMON"—Sets date range based on the run month.
- "**PREDAY"—Sets date range based on the run date, less one day.
- "**PREWEK"—Sets date range based on the previous week (Sun–Sat).
- "**PREMON"—Sets date range based on the previous month.

Comments

This property overrides:

- The date parameters specified in the AddEntry or AddEntries method.
- The date specified by the DateStart and DateEnd properties.

DateStart

Returns or sets a default date value that specifies the starting date field to be written to the records in the CSR file.

Syntax

object.DateStart [=*value*]

Parameter

Value

A date value specifying the date start value of the CIMS Server Resource Record.

Comments

This property is overridden if the DateKeyword property is specified.

DebugMessage

Returns a string value that contains detailed internal counters about the aggregation run.

Syntax

object.DebugMessage

Parameters

None.

Comments

None.

LastErrorMessage

Returns a string value description of the error message generated by the last method or property call.

Syntax

object.LastErrorMessage

Parameters

None.

Comments

If no errors are generated by the last method or property call, an empty string is returned.

MemoryMinimum

Returns or sets an integer value that specifies the minimum amount of memory in megabytes that CIMS Aggregation Engine will use to store aggregates.

Syntax

object.MemoryMinimum [=*value*]

Parameter

Value

An integer value specifying the minimum amount of memory used to store aggregates.

Comments

CIMS Aggregation Engine will use the amount of memory specified by the minimum memory value even if the amount of physical memory available is less than this value. This property is useful when other processes consume all available physical memory. By specifying a minimum, CIMS Aggregation Engine might be able to force the release of some physical memory for its use.

The aggregation engine will request that operating system set the process working set size to be in the range set by the MemoryMinimum and MemoryMaximum properties. The process working set is the set of memory pages currently visible to the process in physical memory. These pages are resident and available for use without triggering a page fault.

MemoryMaximum

Returns or sets an integer value that specifies the maximum amount of memory in megabytes that CIMS Aggregation Engine will use to store aggregates.

Syntax

`object.MemoryMaximum [=value]`

Parameter

Value

An integer value specifying the maximum amount of memory used to store aggregates.

Comments

The aggregation engine will request that operating system set the process working set size to be in the range set by the `MemoryMinimum` and `MemoryMaximum` properties. The process working set is the set of memory pages currently visible to the process in physical memory. These pages are resident and available for use without triggering a page fault.

OutputFileName

Returns or sets a string value that specifies the output filename of the CSR file to be written.

Syntax

`object.OutputFileName [=value]`

Parameter

Value

A full path and filename that determines where the CSR file will be written.

Comments

The parameter must be specified. There is no default.

ResultsMessage

Returns a string value that contains detailed internal counters about the aggregation run.

Syntax

`object.ResultsMessage`

Parameters

None.

Comments

None.

WorkFilePath

Returns or sets a string that specifies a complete file system path where the work files, if required, will be written.

Syntax

object.WorkFilePath [=*value*]

Parameter

Value

A string specifying the complete file system path where the work files will be written.

Comments

The default is to use the path specified by the TEMP environment variable. If the TEMP environment variable is not defined, the current directory is used.

Work files are not always generated. Work files are generated when all of the aggregates will not fit into memory.

TypedEngine and ScriptingEngine Interface Methods

AddEntry

Adds a list of identifier values and resource values to an aggregate.

Syntax

```
object.AddEntry(ByRef IdentifierValueList() As String, _
                ByRef ResourceValueList() As String, _
                Optional ByVal DateStart As Date, _
                Optional ByVal DateEnd As Date) As Long
```

Parameters

IdentifierValueList

Provides a list of identifier value strings (it cannot be an empty a list). The number of entries must match the number of entries specified in the `DefineIdentifier` method.

ResourceValueList

Provides a list of numeric resource values (it cannot be an empty a list). The number of entries must match the number of entries specified in the `DefineResource` method.

DateStart

An optional parameter that specifies the starting date for this entry. If no entry is specified, the default specified with the `DateStart` or `DateKeyword` property is used.

DateEnd

An optional parameter that specifies the ending date for this entry. If no entry is specified, the default specified with the `DateEnd` or `DateKeyword` property is used.

Return Value

Returns a CIMS result code indicating whether the entry specified was added successfully. The CIMS result codes are:

- Successful = `0`
- Warning = `8`
- Error = `16`

Comments

The `DateStart` and `DateEnd` parameters are overridden if:

- The `DateKeyword` property is specified.
- The `DateStart` and `DateEnd` properties are specified.

To perform date aggregation, the `DateStart` and `DateEnd` parameter values must be specified.

The identifier value list is matched in the same order as identifier names are defined. The resource value list is matched in the same order as rate codes are defined.

AddEntries

Batches several calls to the `AddEntry` method into a single call resulting in lower processing overhead.

Syntax

```
object.AddEntries(ByVal NumberOfEntries As Variant, _
                  ByRef IdentifierValueList() As Variant, _
                  ByRef ResourceValueList() As Variant, _
                  Optional ByRef DateStartList As Variant, _
                  Optional ByRef DateEndList As Variant) _
                  As Long
```

Parameters

NumberOfEntries

Specifies the number of valid entries contained in the identifier value string lists.

IdentifierValueList

Provides a list of identifier value strings (it cannot be an empty a list). The number of identifier values must match the number of entries specified in the `DefineIdentifier` method.

The array must be declared with the number of identifier values first, followed by the number of entries in the list. For example, if there are 1000 entries each with 3 identifier values, the array is declared in VBScript as (`2, 999`). (Arrays in VBScript begin counting at 0).

ResourceValueList

Provides a list of numeric resource values (it cannot be an empty a list). The number of resource values must match the number of entries specified in the `DefineResource` method.

The array must be declared with the number of resource values first, followed by the number of entries in the list. For example, if there are 1000 entries each with 3 resource values, the array is declared in VBScript as (`2, 999`).

DateStartList

An optional parameter that specifies the starting date list. If no entry is specified, the default specified with the `DateStart` or `DateKeyword` property is used. If a list is specified, all entries in the list must contain a valid date.

DateEndList

An optional parameter that specifies the ending date list. If no entry is specified, the default specified with the `DateEnd` or `DateKeyword` property is used. If a list is specified, all entries in the list must contain a valid date.

Return Value

Returns a CIMS result code indicating whether all of the entries specified were added successfully. The CIMS result codes are:

- Successful = `0`
- Warning = `8`
- Error = `16`

Comments

This method is currently implemented only in the `ScriptingEngine` interface.

The `DateStartList` and `DateEndList` parameters are overridden if:

- The `DateKeyword` property is specified.
- The `DateStart` and `DateEnd` properties are specified.

To perform date aggregation, the `DateStartList` and `DateEndList` parameter values must be specified.

The identifier value list is matched in the same order as identifier names are defined. The resource value list is matched in the same order as rate codes are defined.

ClearIdentifierList

Clears the internal list of identifier names.

Syntax

```
object.ClearIdentifierList()
```

Parameters

None.

Comments

None.

ClearResourceList

Clears the internal list of rate codes.

Syntax

```
object.ClearResourceList()
```

Parameters

None.

Comments

None.

DefineIdentifier

Adds an identifier name to an internal list of identifier names.

Syntax

```
object.DefineIdentifier(ByVal IdentifierName As String)
```

Parameter

IdentifierName

Provides a string value containing an identifier name.

Comments

Identifiers names must be defined in the same order that the identifier values appear in the `AddEntry` or `AddEntries` method.

There must be at least one identifier name defined.

DefineResource

Adds a rate code to an internal list of rate codes.

Syntax

```
object.DefineResource(ByVal RateCode As String,
                      Optional ByVal ResourceConversionFactor As Double
                      Optional ByVal DecimalPositions As Long)
```

Parameters

RateCode

Provides a string value containing a rate code.

ResourceConversionFactor

An optional parameter that divides the incoming resource values passed to the `AddEntry` or `AddEntries` method by a double value. The default value is `1`. This is an optional parameter.

DecimalPositions

An optional parameter that specifies the number of decimal digits that resource values are rounded to. Zero rounds to a whole number. By default, the values are not rounded.

Rounding is based on 5. For example, a resource value of 3.5 rounds to 4 if 0 is specified for the decimal digits. A value of 5.53 rounds to 5.5 if a decimal digit of 1 is specified.

Comments

Rate codes must be defined in the same order that the resource values appear in the `AddEntry` or `AddEntries` method.

There must be at least one rate code defined.

If a resource conversion factor of `1` is specified, then no division of resource values takes place.

DefineResourceRecordHeader

Specifies the resource record header that the records in the CSR file should use.

Syntax

```
object.DefineResourceRecordHeader(ByVal ResourceRecordHeader As String)
```

Parameter

ResourceRecordHeader

Provides a string value containing the resource record header to be used for records generated by the `AddEntry` or `AddEntries` method.

Comments

This method can be set once for all records, called once for each record, or called as needed. The default value is `NONE`.

Initialize

Initializes the aggregation object.

Syntax

```
object.Initialize(Optional ByVal MaxEntries as Long) As Boolean
```

Parameter

MaxEntries

An optional parameter that specifies how may aggregates to store in memory. The default is to store as many aggregates as will fit in memory.

Return Value.

Returns `True` if initialization is successful. Returns `False` if otherwise.

Comments

This method should be the first call made to CIMS Aggregation Engine. It resets all properties to their default values and resets the internal state of the object.

WriteResourceFile

Releases all aggregation records to the CSR file.

Syntax

```
object.WriteResourceFile() As Boolean
```

Parameters.

None.

Return Value.

Returns `True` if the CSR file was written successfully. Returns `False` if otherwise.

Comments

This method must be called for the CSR file to be written. If all aggregates do not fit into memory, this method initiates work file processing. When the method returns, full aggregation of the input file has been completed and the CSR file has been written.

ExceptionFile Interface

ExceptionFile Interface Properties

ExceptionCount

Returns a count of the number of exception records written so far.

Syntax

object.ExceptionCount

Parameters

None.

Comments

None.

FileName

Returns or sets the name of the exception file.

Syntax

object.FileName [=*value*]

Parameters

Value

A string value specifying the full path and filename of the exception file.

Comments

The default filename is CIMSExceptionFile.txt.

MaxExceptions

Returns or sets the maximum number of exception entries.

Syntax

object.MaxExceptions [=*value*]

Parameters

Value

A long value specifying the maximum number of entries that can be written to the exception file.

Comments

To allow an unlimited number of exception entries, set this property to -1.

The default is to allow an unlimited number of exception entries.

Once the maximum number of exceptions has been reached, no more entries are written to the exception file.

MaxExceptionsReached

Returns a Boolean value indicating whether the maximum number of exceptions generated by the AddException method exceed the number specified by the MaxException property.

Syntax

object.MaxExceptionsReached

Parameters

None.

Comments

If an unlimited number of exception entries is allowed (the default), then the return value is always False.

ExceptionFileInterface Method

AddException

Adds an exception record to an exception file.

Syntax

```
object.AddException(ByVal Value as Long) As String
```

Parameter

Value

A string value that contains the source record that could not be processed.

Return Value

Returns `True` if the source record string could be added to the exception file. Returns `False` otherwise.

Comments

An exception file name must be specified by setting the `FileName` property.

The CSR file is closed when the script exits or the object goes out of scope.

If no exceptions are generated, the exception file is not created.

B

CIMS Processing Engine API

About CIMS Processing Engine

CIMS Processing Engine is composed of COM objects. Each object performs particular functions in the data processing cycle. Table 2-1 describes the CIMS Processing Engine COM objects in the order that they appear in the processing cycle. (For a complete description of the data processing cycle, refer to the *CIMS Server Administrator's Guide*.)

The CIMS Processing Engine COM objects are called by the `ProcCIMS.wsf` script (see page 2-17).

COM Object	Description
`CIMSACCT.dll`	Performs account code conversion, shift determination, date selection, and identifier extraction on the usage data and produces the CIMSACCT Detail File containing records that are properly formatted for input into CIMSBILL. See *CIMSACCT.DLL* on page B-3 for the `CIMSACCT.dll` properties and methods.
`CIMSSORT.dll`	Converts the CIMSACCT Detail File and produces a sorted version that is ready to be processed by CIMSBILL. CIMSSORT also combines multiple files into one file—you can use CIMSSORT to combine CIMSACCT output from different data collectors into one file.
`CIMSBILL.dll`	Processes the sorted CIMSACCT Detail File from CIMSSORT and performs shift processing, CPU normalization, proration, and include/exclude processing and creates the CIMSBILL Detail and CIMS Summary Files that contain the billing information used to generate invoices and reports.
`CIMSAdminLib.dll`	Contains the class `CCIMSCBSLoad` that loads the output files from CIMSACCT and CIMSBILL into the database.
`CIMSResr1.dll`	Supports CIMSACCT and CIMSBILL.

Table 2-1 • CIMS Processing Engine Components

CIMSACCT.DLL

The CIMSACCT.dll contains the CIMSACCT object. The following sections describe the object properties and Execute method.

CIMSACCT Properties

The CIMSACCT properties are presented in the order in which they appear in the ProcCIMS.wsf script. These properties are optional or required as noted.

- CREATEDBINF **(optional).** This property instructs CIMSACCT to create the database information file (ODBCINF.txt) from data source selected for the process. If this property is not set, CIMSACCT uses the database information that is currently present in the ODBCINF.txt file, which may not be the same as the data source selected for the process.
- Path **(required).** This property specifies the directory path for all CIMSACCT input/output files.
- DetailFile **(required).** This property is used to set the file name of the output CIMSACCT Detail File (excluding the path).
- AccCodeConvTable **(optional).** This property is used to set the file name of the input account code conversion table (excluding the path name). This is optional unless the ACCOUNT CODE CONVERSION statement is included in the CIMSACCT input control file. For more information about the input control file and control statements, refer to the *CIMS Server Administrator's Guide*.
- InputFile **(required).** This property is used to set the file name of the input record file (excluding the path).
- ResultsFile **(optional).** This property is used to set the file name of the output CIMSACCT Results File (excluding the path). If it is not set, then CIMSACCT produces a warning message that states that the file is not found.
- ControlFile **(required).** This property is used to set the file name of the input CIMSACCT control file (excluding the path).
- MessageFile **(optional).** This property is used to set the file name of the output CIMSACCT Message File (excluding the path name). If this is not set, then CIMSACCT produces a warning message that states that the file is not found.
- ExceptionFile **(optional).** This property is used to set the file name of the output exception file (excluding the path name). This property is optional, unless the EXCEPTION FILE PROCESSING ON statement is included in the CIMSACCT input control file.
- IdentFile **(required).** This property is used to set the file name of the output CIMS Ident File (excluding the path).
- DBDataSource **(required).** This property is used to set the ODBC data source name as defined in the ODBCINF.txt file.

- `DBUser` **(optional).** This property is used to set the user ID to use to log onto the database. You can also specify this in the `ODBCINF.txt` file.
- `DBPass` **(optional).**This property is used to set the password to use to log onto the database. You can also specify this in the `ODBCINF.txt` file.

CIMSACCT Method

The only CIMSACCT method is `EXECUTE`. The `EXECUTE` method returns one of the following return codes:

`0`	Execution ended with no errors or warnings
`4` or `8`	Execution ended with warning messages. Check the CIMS Event Viewer and CIMSACCT Message File in CIMS Server Administrator.
`16`	Execution ended with errors—processing stopped. Check the CIMS Event Viewer and CIMSACCT Message File in CIMS Server Administrator.

CIMSACCT Code Example

The `ProcCIMS.wsf` script creates the CIMSACCT object, sets the object properties, and executes CIMSACCT as shown in the following example:

```
Set oCIMSACCT = CreateObject("CIMSACCT.CIMSACCT")
oCIMSACCT.CREATEDBINF = "Y"
oCIMSACCT.Path = "e:\processes\MSExchange\"
oCIMSACCT.DetailFile = "detail.txt"
oCIMSACCT.AccCodeConvTable = "accttable.txt"
oCIMSACCT.InputFile = "CurrentCSR.txt"
oCIMSACCT.ResultsFile = "acctresults.txt"
oCIMSACCT.ControlFile = "acctcntl.txt"
oCIMSACCT.MessageFile = "acctmsg.txt"
oCIMSACCT.ExceptionFile = "exception.txt"
oCIMSACCT.IdentFile = "ident.txt"
oCIMSACCT.DBDataSource = "CIMSServer"
oCIMSACCT.DBUser = ""
oCIMSACCT.DBPass = ""
intRC = oCIMSACCT.Execute
```

CIMSSORT.DLL

This DLL file contains two objects:

- CTextFileSort. Converts the CIMSACCT Detail File(s) and produces a sorted version that is ready to be processed by CIMSBILL.
- CTextFileMergeSort. Sorts multiple files into one file. For example, you can combine CIMSACCT output from three different data input types into one file.

The `ProcCIMS.wsf` script does not create the CTextFileMergeSort object by default. For assistance creating and executing CTextFileMergeSort, contact CIMS Lab (see *Chapter 13, Contacting Technical Support*).

The following sections describe the CIMSSORT properties and `Sort` method.

CIMSSORT Properties

The CIMSSORT properties are presented in the order in which they appear in the `ProcCIMS.wsf` script. These properties are required.

- `InputFileName` **(CTextFileSort only).** The input CIMSACCT Detail File with full path.
- `OutputFileName` **(CTextFileSort only).** The output sorted CIMSACCT Detail File with full path.
- `KeyOffset`. The account code start position in the CIMSACCT Detail File. The account code starts in position 163.
- `KeyLength`. The account code length. The maximum account code is length is 128 bytes.

CIMSSORT Method

The only CIMSSort method is `Sort`. The `Sort` method returns `TRUE` or `FALSE` depending on whether it is successful.

CIMSSORT Code Example

The `ProcCIMS.wsf` script creates the CTextFileSort object, sets the object properties, and executes CTextFileSort as shown in the following example:

```
Set oCIMSsort = CreateObject("CIMSSort.CTextFileSort")
oCIMSsort.InputFileName  = strSortIn
oCIMSsort.OutputFileName = strProcessesFolder & strSortOut
oCIMSsort.KeyOffset = ACCOUNT_CODE_START_POSITION
oCIMSsort.KeyLength = ACCOUNT_CODE_START_LENGTH
If (oCIMSsort.Sort) Then
    WScript.Echo("SORT Finished...")
    'WScript.Echo("SORT Stats:" & oCIMSsort.SortStats)
Else
    WScript.Echo("Sort Failed.....")
    WScript.Quit(Fail)
End If
Set oCIMSsort = Nothing
```

CIMSBILL.DLL

The `CIMSBILL.dll` contains the CIMSBILL object. The following sections describe the object properties and `Execute` method.

CIMSBILL Properties

The CIMSBILL properties are presented in the order in which they appear in the `ProcCIMS.wsf` script. These properties are optional or required as noted.

- `Path` **(required).** This property specifies the directory path for all CIMSBILL input/output files.
- `SummaryFile` **(required).** This property is used to set the file name of the output CIMS Summary File (excluding the path).
- `DetailFileIn` **(required).** This property is used to set the file name of the input CIMSACCT or CIMSBILL Detail or CIMS Summary File (excluding the path).
- `DetailFileOut` **(required).** This property is used to set the file name of the output CIMSBILL Detail File (excluding the path).
- `ResultsFile` **(optional).** This property is used to set the file name of the output CIMSBILL Results File (excluding the path). If it is not set, then CIMSBILL produces a warning message that states that the file is not found.
- `ControlFile` **(required).** This property is used to set the file name of the input CIMSBILL control file (excluding the path).
- `MessageFile` **(optional).** This property is used to set the file name of the output CIMSBILL Message File (excluding the path name). If this is not set, then CIMSBILL produces a warning message that states that the file is not found.
- `MultTableFile` **(optional).** This property is used to set the file name of the input proration table (excluding the path). This property is optional, unless the `PRORATE MONEY` or `PRORATE RESOURCES` statement is included in input CIMSBILL control file.
- `DBDataSource` **(required).** This property is used to set the ODBC data source name as defined in the `ODBCINF.txt` file.
- `DBUser` **(optional).** This property is used to set the user ID to use to log onto the database. You can also specify this in the `.ODBCINF.txt` file.
- `DBPass` **(optional).**This property is used to set the password to use to log onto the database. You can also specify this in the `ODBCINF.txt` file.

CIMSBILL Method

The only CIMSBILL method is `EXECUTE`. The `EXECUTE` method returns one of the following return codes:

`0`	Execution ended with no errors or warnings
`4` or `8`	Execution ended with warning messages. Check the CIMS Event Viewer and CIMSBILL Message File in CIMS Server Administrator.
`16`	Execution ended with errors—processing stopped. Check the CIMS Event Viewer and CIMSBILL Message File in CIMS Server Administrator.

CIMSBILL Code Example

The `ProcCIMS.wsf` script creates the CIMSBILL object, sets the object properties, and executes CIMSBILL as follows:

```
Set oCIMSBILL = CreateObject("CIMSBILL.CIMSBILL")
oCIMSBILL.Path = "e:\processes\MSExchange\"
oCIMSBILL.SummaryFile = "BillSummary.txt"
oCIMSBILL.DetailFileIn = "SortOut.txt"
oCIMSBILL.DetailFileOut = "BillDetail.txt"
oCIMSBILL.ResultsFile = "BillResults.txt"
oCIMSBILL.ControlFile = "BillCntl.txt"
oCIMSBILL.MessageFile = "BillMsg.txt"
oCIMSBILL.MultTableFile = "prorate.txt"
oCIMSBILL.DBDataSource = "CIMSServer"
oCIMSBILL.DBUser = ""
oCIMSBILL.DBPass = ""
If Len(strLogDate) > 0 Then
    WScript.Echo "Overriding Report Date with: " & strLogDate
    oCIMSBILL.ReportDate = strLogDate
End If
intRC = oCIMSBILL.Execute
```

CimsAdminLib.DLL

The `CIMSAdminLib.dll` is the central controlling library for all Windows-based CIMS Server functions. It contains all the COM objects used to manipulate CIMS data including the class `CCIMSCBSLoad` that loads the output files from CIMSACCT and CIMSBILL into the database.

For more information about CIMSAdminLib, please contact CIMS Lab ().

C

Running Batch Scripts

Note • The scripts `Nightly.bat` and `Monthly.bat` are shipped as `SampleNightly.bat` and `SampleMonthly.bat`. You need to rename these scripts `Nightly.bat` and `Monthly.bat` (or choose other names) so that the files are not overwritten when you upgrade to a new version of CIMS Data Collectors. These scripts are referred to as `Nightly.bat` and `Monthly.bat` in this guide.

You can run the `Nightly.bat` or `Monthly.bat` script directly from the command prompt or you can use Windows Scheduled Tasks to schedule the script to run automatically.

To run the batch script from the command prompt:

At the `C:\Program Files\CIMSLab\SCRIPTS>` prompt, type `Nightly` or `Monthly` followed by the LogDate parameter (`preday`, `curday`, etc.) and press <Enter>. For example, `C:\Program Files\CIMSLab\SCRIPTS>Nightly preday` processes usage metering files produced on the previous day.

Note that the path for the `Scripts` folder may differ depending on your installation.

To run batch scripts from Scheduled Tasks:

Note • These instructions are for Windows 2000 Server.

1. Open Scheduled Tasks.
2. Double-click **Add Scheduled Tasks**.
3. The **Scheduled Task Wizard** appears. Click **Next**.
4. Click **Browse**, select the `Nightly.bat` or `Monthly.bat` script, and then click **Next**.
5. Type a name for the task or accept the default and click the schedule for the task. Click **Next**.
6. Select the time and day to start the task, and then click **Next**.
7. Type the password for the user account on which you want the the scheduled task to run. The password cannot be blank. Click **Next**.

8. Select the **Open advanced properties for this task when I click Finish** check box, and then click **Finish**.
9. In the **Task** tab, type the LogDate parameter in the **Run** box as shown in the following example:

   ```
   "C:\Program Files\CIMSLab\SCRIPTS\Nightly.bat" preday
   ```

   Where `preday` specifies process usage metering files produced on the previous day.
10. Click **OK**.

The task appears in the Scheduled Task list. To run the `Nightly.bat` or `Monthly.bat` script immediately, right-click the task, and then click **Run**. For more information about Scheduled Tasks, refer to the Microsoft documentation.

D

Adding Data Sources

Note • This appendix is applicable only if you are using a Microsoft SQL Server collector or the DBSpace collector.

If you are collecting database data, you need to create an ODBC data sources for the databases in the following situations:

1. If you are using a CIMS Data Collector for Microsoft SQL Server (2000 or 7.0) and are "pulling" trace files from a database or databases into the trace folder on the collector server.
2. If you are using the CIMS Data Collector for database size, DBSpace.

To create a data source, use Data Sources (ODBC). You can reach this tool from the Windows Control Panel or CIMS Server Administrator.

To open from Windows Control Panel:

1. Click **Start** ▸ **Settings** ▸ **Control Panel**.
2. Double-click **Administrative Tools**, and then double-click **Data Sources (ODBC)**.

To open from CIMS Server Administrator:

1. In the CIMS Server Administrator main window, click **Select ODBC Data Source**. The Select ODBC Data Source dialog box appears.
2. Click **ODBC Data Source Administrator**.

To add the data source:

1. In the ODBC Data Source Administrator dialog box, click the **System DSN** tab.
2. Click **Add** and follow the instructions provided by the DSN creation wizard. Consult your SQL Server DBA to determine the settings that you should select in the wizard.
3. When the wizard completes, click **Test Data Source**, and then click **OK** until the Select ODBC Data Source dialog box closes.

To confirm that the new data source has been added, click **Select ODBC Data Source** again. The data source should appear in the **System Data Sources** list.

Glossary

CIMS Aggregation Engine • CIMS Aggregation Engine is a COM object that aggregates the records within the usage metering file by identifier values and produces a CIMS Server Resource File. Because the data in the usage metering file has been aggregated, the resulting CIMS Server Resource File requires less processing time. *See also identifier.*

CIMS Conversion Builder • CIMS Conversion Builder is a GUI application that you can use to create definition files for the usage metering files. These definition files are fed into CIMS Conversion Engine. CIMS Conversion Builder is used only with CIMS Universal Collector. *See also CIMS Conversion Engine, CIMS Universal Data Collector, and definition file.*

CIMS Conversion Engine • CIMS Conversion Engine is a COM object that enables usage metering files to be processed by CIMS Server. CIMS Conversion Engine reformats the data in the files into CIMS Server Resource Files. CIMS Conversion Engine is used only with CIMS Universal Collector. *See also CIMS Universal Data Collector.*

CIMS Processing Engine • CIMS Processing Engine is composed of COM objects that process the CIMS Server Resource Files created by CIMS Aggregation Engine or CIMS Conversion Builder and load the output into the CIMS Server database.

CIMS Server Resource (CSR) File • The resource file that contains the data that is input into CIMS Server. The CIMS Server Resource File contains CIMS Server Resource Records. These records are comma-delimited and can contain a very large number of resource identifiers and resources. *See also identifier and rate code.*

CIMS Universal Data Collector • A universal data collection process for applications that do not have a specific CIMS Data Collector.

COM • Acronym for Component Object Model. A specification developed by Microsoft for building software components that can be assembled into programs or add functionality to existing programs running on Microsoft Windows platforms.

CPU • Acronym for central processing unit. The computational and control unit of a computer.

CSR File • *See CIMS Server Resource (CSR) File.*

DLL • Acronym for dynamic-link library. A module that contain functions and data that can be used by another module (application or DLL).

definition file • The definition file defines the format of the usage metering file as well as the format of the output file to be produced by CIMS Conversion Engine.

identifier • In the CIMS Server Resource Record, a unique key that denotes the source of a resource that has been consumed. Examples include device name, server name, system ID, phone number, user ID, state code or building number. A consumed resource can have one to many identifiers.

ODBC • Acronym for Open Database Connectivity. An interface providing a common language for database access.

process • An executable application, such as Microsoft Word, or a service such as MSTask.

process definition folder • A folder that contains the files required to process usage data from a particular source such as a database, operating system, or application.

rate code • A rate code represents the resource units being reported (for example, CPU time, transactions processed or lines printed). The rate code includes the value for the resource and other rate processing information.

UNC • Acronym for Universal Naming Convention. A file naming system beginning with two backslashes (\\) that indicates that the resource exists on a network computer. The syntax is `\\Servername\Sharename`.

usage metering file • A file that contains usage data for an application. For example, a log file.

UTC • Acronym for Universal Time Coordinate. A world-wide standard for time and date. Formerly know as Greenwich Mean Time (GMT). Also referred to as Zulu time, universal time, and world time.

VBScript • Abbreviation for Visual Basic, Scripting Edition. A subset of the Visual Basic for Applications programming language, optimized for Web-related programming.

Windows Script Component • A script tool for creating COM components. Script component files are indicated by the extension `.wsc`. These files are XML (Extensible Markup Language) files that contain information about the COM component. *See also COM*.

Windows Script File • A Windows script (`.wsf`) file is a text document containing Extensible Markup Language (XML) code. Windows script files are not engine-specific and can contain script from any Windows Script compatible scripting engine.

Windows Script Host (WSH) • A language-independent scripting host for Windows Script-compatible scripting engines. WSH acts as a host for scripts—it makes objects and services available for the script and provides a set of guidelines within which the script is executed.

XML • Acronym for Extensible Markup Language. A meta-markup language that provides a format for describing structured data. XML allows for more precise declarations of content and more meaningful search results across multiple platforms.

Index

D

E

F

I

T

U

V

W

X