The Mainframe Software Partner
For The Next 50 Years

Enterprise Common Components

Installation and Customization Guide
MVS Version

Release 16.05

Global technical support contact information is available from our Customer Support Web Site:

https://go.compuware.com

Doc. JUL2016
August 30, 2017.

Contents

Introduction

Important:

Compuware mainframe installs, configurations, and customizations have been standardized to be consistent between products, releases, and across LPARs.

ALL 17.02 Compuware mainframe products have been standardized to use a common PARMLIB for specifying configuration and customization parameter values, serviced through the Compuware Mainframe Services Controller (CMSC).

ALL 17.02 Compuware mainframe products require ECC 16.05 including all maintenance.

Pre-16.05 versions of the ECC components (LMS 4.0, HCI 3.0, and CSS 9.0) will not be supported by Compuware mainframe products 17.02 or greater.

ECC Introduction

Important:

Beginning with release 16.05 (released January 2016), Enterprise Common Components (ECC) consists of a single FMID operating from one authorized load library and one non-authorized load library. ECC is one product containing the following functional components:

- Compuware Mainframe Services Controller (CMSC)
- License Management System (LMS)
- Compuware Shared Services (CSS)
- Host Communications Interface (HCI)
- Base Services

Note: The Compuware Mainframe Services Controller (CMSC) address space provides common parameter library services.

- Beginning with ECC 16.05, License Management runs in the CMSC address space. The CMSC can also automatically start HCI address spaces.

  The HCI now uses a single simplified parameter member that supports all functionality in the HCI including Topaz, Strobe, and DevEnterprise.
- Beginning with 17.02, Compuware mainframe products are configured through a common PARMLIB, serviced by the CMSC address space.

Common PARMLIB for Configurations and Customizations for 17.02 Mainframe Products

Compuware's mainframe products release 17.02 or greater use a common PARMLIB for configurations, serviced through the Compuware Mainframe Services Controller (CMSC) address space (see Chapter 5, "Compuware Mainframe Services Controller (CMSC)" for more information). Parameter values are stored in a PDS or concatenation of PDS

libraries. The parm members are human readable and editable, which allows administrators to view or change values using an operator command. z/OS System Symbolics are supported by common PARMLIB members, which can be extremely helpful in simplifying, for example, multiple LPAR deployments.

Being a common PARMLIB library to all Compuware mainframe products (17.02 and above), similar product information can be expressed to more than one Compuware product. For example, Strobe information can be defined for both Strobe and another product, like HCI.

Users can also specify different parm members for different invocations of individual products. For example: HCIxx where the xx identifies which PARMLIB member is to be used.

Manual Structure

This *Enterprise Common Components Installation and Customization Guide* is composed of the following chapters:

Chapter 1, "ECC Overview": A description of the individual components that make up Enterprise Common Components with overview information on each of the parts. These include: ECC, CMSC,CSS, LMS, and HCI.

Chapter 2, "Preparing for ECC Installation": Critical information for planning your installation of ECC including the LMS, CSS, and HCI components. This chapter also lists which components are required based on your Compuware product, and also lists the libraries that are created during installation with SMP/E.

Chapter 3, "Receive From Network (RFN) Installation": Step-by-step instructions for SMP/E installing Compuware Shared Services, License Management System, and Base Services/HCI using Receive From Network (RFN) procedures.

Chapter 4, "Applying Maintenance": Mandatory step-by-step instructions for applying maintenance to your installation of ECC.

Chapter 5, "Compuware Mainframe Services Controller (CMSC)": Instructions for setting up and controlling CMSC address space for use with the Compuware products licensed by your organization.

Chapter 6, "LMS Customization": Step-by-step instructions for setting up and customizing LMS for use with the Compuware products licensed by your organization.

Chapter 7, "HCI Customization": contains important information that can help you install the mainframe software needed to support host communication from the mainframe to certain Compuware products.

Chapter 8, "CSS Customization": Step-by-step instructions for customizing your installation of CSS.

Chapter 9, "Verification of ECC Component Customization": Procedures to verify the installation and customizations of ECC Components.

Intended Audience

This manual is intended for installers and application programmers. It is only to be used for installing the MVS/ESA, OS/390, and z/OS versions of Compuware products.

Related Publications

For more information about ECC products refer to the Compuware documentation on FrontLine (**https://go.compuware.com**). For more information on compiling COBOL, PL/I, Assembler, or C/C++ programs, refer to the appropriate IBM manuals.

FrontLine Support Web Site

Access online technical support for Compuware products through our FrontLine support Web site. View or download documentation, frequently asked questions, and product fixes, or e-mail Compuware with questions or comments. To access FrontLine, you must first register and obtain a password at **https://go.compuware.com**.

Online Documentation

The following electronic formats are available on **https://go.compuware.com**:

- View PDF files with the free Adobe Reader, available at http://www.adobe.com.
- View HTML files with any standard Web browser.

World Wide Web

Compuware's site on the World Wide Web provides information about Compuware and its products. The address is **https://go.compuware.com**.

Terminology

The following terms are defined as they are used in this manual:

- **DDIO (Dump Dataset Input Output) file** is used generically to mean any of the following types of files created by the CSS utilities:
  - Report File
  - Source Listing File
  - Transaction Databases and Shared Directories
  - Report Shared Directory and Database
  - Source Listing Shared Directory
  - Source Listing Database.

  For a list of Compuware products that use/require these types of DDIO files, see Chapter 5, "CSS Customization".
- **Member** is used to identify an individual diagnostic report in an Abend-AID report file, a source listing in a source listing file, or a profile in a profile file.
- **Entry** is used to identify an individual transaction diagnostic report in a transaction database for Abend-AID for CICS.

Getting Help

Compuware continually strives to improve our software products and documentation. Feedback from our customers helps us to maintain our quality standards.

While the software should execute according to documented specifications, there are times when problems do occur. If problems arise, check the Compuware manuals for assistance. You may also need to consult your site's technical representative for Compuware assistance.

If you have attempted to solve the problem using the resources listed above and the difficulty persists, contact the Compuware Customer Support Hotline for assistance. Customer support is also available via our FrontLine Support web site as described in "Related Publications".

Please have the following information available before calling. This information will assist the Customer Support personnel in determining the exact cause of your problem in a timely manner.

Note: If you encounter a CSS problem when compiling your program, Compuware Customer Support staff may need additional information. You can obtain this information by using the CSS Problem Documentation Utility. Refer to the "Using the Problem Documentation Utility" appendix in the *Compuware Shared Services User/Reference Guide* for complete instructions.

1. Use the following sample JCL members to identify the PTFs and/or APARs that have been applied to the ECC libraries:
   - CXLPTFS or CXLZAPS in the ECC SLCXCNTL library

   These jobs (with the exception of CXLZAPS) will produce a detailed SMP/E report of PTFs applied to the above components. The CXLZAPS job will produce a more compact/summarized version of the ECC PTF listing.
2. Determine the release number of the ECC-enabled product.
3. For CSS problems, identify the compiler language and the release number being processed.
4. Identify the CICS release in use if the problem involves a CICS product.
5. Identify the operating system release to help determine operating system dependencies.
6. If an abend occurs, note the displacement and the module in which the interrupt occurs.
7. Note the sequence of issued commands that resulted in the abend, the data type, and the programming language being used.
8. Locate your Compuware customer number.

Compuware Customer Support
http://go.compuware.com

Chapter 1.
ECC Overview

This chapter explains the composition of Compuware's Enterprise Common Components (ECC) and provides overview information on each of its component parts.

What is ECC?

Beginning with Release 16.05, Enterprise Common Components represents a single install image for the following Compuware facilities:

- Compuware Mainframe Services Controller (CMSC)
- License Management System (LMS)
- Compuware Shared Services (CSS)
- Host Communications Interface (HCI)
- Base Services

Note: The Compuware Installer is included as part of the ECC EP media and is only used to facilitate SMP/E installation of Compuware mainframe products. It is not part of the Enterprise Common Components single install image.

All components in the ECC single install image are on the same release level. Compuware recommends that you install and activate all of the components at the same time to avoid compatibility issues between different release levels.

CMSC, CSS, LMS, HCI, and Base Services are all installed with a single install image and a single FMID. Which parts of ECC you need to activate and configure depends on which Compuware products are licensed by your organization. All Compuware mainframe products require the License Management System. Many Compuware products use Compuware Shared Services to provide crucial core functionality. Abend-AID, Abend-AID for CICS, Strobe, DevEnterprise, Hiperstation, Compuware Workbench, and Topaz Workbench use the HCI component. Abend-AID, Abend-AID for CICS, and CMSC use Base Services. Table 2-1 on page 2-1 in Chapter 2, "Preparing for ECC Installation" makes it easy to determine exactly which parts of ECC you need to install.

Each of the components are installed with the ECC single install image using SMP/E and can be configured individually, depending on your site's requirements:

- **Compuware Mainframe Services Controller (CMSC):** The Compuware Mainframe Services Controller (CMSC) address space is a centralized facility providing license management and common parameter library services. The CMSC also has the ability to automatically start HCI address spaces. HCI address spaces started by the CMSC will be automatically restarted in the event of a failure. Operator commands are provided to control the Compuware License Management System. Commands are also provided to control HCI address spaces automatically started by the CMSC. Compuware intends to add additional functionality to the CMSC in the future.

  As the service provider for the new Compuware common parameter library feature, the CMSC reads the user's parameter libraries, stores parameters for Compuware products in memory, and initializes the parameter retrieval service used by Compuware products. Operator commands are provided to refresh the parameters in memory in the event of a change to one or more of the parameter library members.

- **License Management System (LMS):** LMS enables you to manage access to all the Compuware products used by your organization. LMS consists of several small

components which allow you to establish, maintain, diagnose, and upgrade your access to the Compuware products licensed by your enterprise:

  – An ISPF License Administration Utility application
  – A runtime environment
  – A program interface to the runtime environment employed by Compuware products.

  LMS License Certificates are English-like text files that are sent to you electronically. You will receive a License Certificate for each new product release under LMS. You can centrally manage all of your organization's License Certificates in a single License File that can then be distributed to various sites.

- **Compuware Shared Services (CSS):** An enterprise-wide tool for sharing compiled program information, CSS is utilized by many Compuware products spanning the Fault Management and Enterprise Testing Solutions product lines. For more information, see "CSS Overview".
- **Host Communications Interface (HCI):** The Host Communications Interface is a facility that provides connectivity between mainframe-based programmer productivity software and peer-node software running on other platforms in a network. The HCI also serves as a server activation facility that is responsible for starting and monitoring application programs that are using the communications facilities.
- **Base Services:** Base Services is an application framework providing batch, server, subsystem and presentation components. Functions provided by Base Services include dynamic allocation, storage management, transaction scheduling, message formatting, operator command routing, recovery and termination management, and screen, menu and message presentation in ISPF, VTAM, CICS and Web browser environments. This framework is used by CMSC, Abend-AID and Abend-AID for CICS.
- **Compuware Installer:** Use the Compuware Installer to facilitate the SMP/E installation of Compuware mainframe products. See the *Compuware Installer—Mainframe Product SMP/E Installation Guide* for more information.

LMS Overview

LMS is a feature of the ECC single install image. LMS enables you to centrally administer Compuware's product License Certificates and manage access to Compuware products at your site.

Beginning with Release 16.05, LMS will execute in the CMSC address space.

LMS Advantages

LMS provides the following advantages:

- Electronically-delivered License Certificates that contain product access parameters in a human-readable text format
- Programs and reports to verify the contents of the License File and LMS runtime environment
- Capability to employ a single centralized Enterprise License File and centralized administration of License Files
- Easy updates to License Certificates and the License File with no disruption to your Compuware products
- No production impact from the testing of LMS runtime systems, installation and validation of new License Certificates, or installation of new or updated LMS software

- Optional E-mail alerts and SMF logging when licensing events occur.

Operating Environment

LMS operates under IBM MVS/ESA Release 4.3 and more current, and all OS/390 and z/OS releases.

The License Management Process

The license management process begins when you acquire a Compuware product through a license agreement, trial agreement, or beta agreement.

Information Gathering

Compuware obtains the relevant information for license management from you at the time of the agreement. Your organization supplies information such as name and the sites and CPUs for which the product is being licensed. Information about the Compuware product licensed, such as product name, release number, and options licensed, is obtained from your Compuware representative.

License Certificate

Upon the completion of an agreement, Compuware creates a License Certificate representing a portion of the information in that agreement. The License Certificate is used by LMS to provide access to Compuware products. The License Certificate is **not** the same thing as a license agreement. You are still responsible for abiding by your license agreement, and although not its primary role, you will find that the LMS can help in that effort.

The License Certificate, LMS, and the Compuware product are all delivered to you by Compuware. Typically, you receive the License Certificate via e-mail, but other methods can be used, if necessary.

Installation

LMS is included as part of the ECC single install image. First, set up your LMS environment, import your License Certificate into a **License File**, execute the **CMSC address space**, and install your Compuware products. From then on, access to your Compuware mainframe products occurs transparently.

After you accomplish the initial LMS installation, you would typically only need to import a new product License Certificate and re-initialize the LMS runtime environment in conjunction with events such as the following:

- obtaining a new release of a product under the terms of your software maintenance agreement
- adding new products through additional license agreements
- amending your agreement to include new options
- changing the CPUs licensed in your original agreement.

CSS Overview

Compuware Shared Services (CSS) is an integral component of many Compuware products spanning the Fault Diagnosis, Interactive Analysis and Debugging, File and Data Management, and Application Performance Measurement product lines. CSS provides storage, retrieval, and maintenance for Abend-AID reports, source listings, and transaction dump information in datasets called DDIO files. Shared directories and their attached databases are also types of DDIO files being utilized for more efficiency in

maintaining information and activity within these files. CSS also provides language processing support for COBOL, PL/I, Assembler, and C.

The following products currently use CSS:

- Abend-AID
- Abend-AID for CICS
- Compuware Workbench/Topaz Workbench
- Strobe
- Xpediter/CICS
- Xpediter/Code Coverage
- Xpediter/TSO and Xpediter/IMS

System Environment

For System Requirement information, see the *Enterprise Common Components Release Notes* available from GO/Frontline (**https://go.compuware.com**).

CSS Components

CSS consists of the following components:

- Compuware common files and utilities
  - DDIO files: Source Listing Files, Report Files, Shared Directories and Associated Report, Transaction, and Source Listing Databases
  - Batch file utilities: CWDDSUTL, CWFXSDUT, CWAASDUT, CWDDLPUT, and CWDDALLU
- Compuware Language Processors (COBOL, Assembler, PL/I, and C)
- Security Exit Program
- Topaz Workbench support

Common Files and Utilities

CSS components include common files and utilities that provide storage, retrieval, and maintenance functions.

DDIO Files

A generic term used to describe the proprietary file access method that stores Abend-AID reports, transaction dump information, and source listings for several Compuware products. DDIO files can be allocated as either VSAM or sequential datasets. Different products may require different types of DDIO files.

CSS Utilities and Batch File Utilities

CSS Utilities walks you step-by-step through CSS language processor functions and DDIO file manipulation using easy-to-use panels.

The batch file utilities (CWDDSUTL, CWAASDUT, CWDDLPUT, CWFXSDUT and CWDDALLU) provide the same functions as CSS Utilities in a batch environment.

For information on the available online utility functions refer to the “CSS Utilities” chapter in the *Compuware Shared Services User/Reference Guide*. For information on the available batch utility commands and parameters for CWDDALLU or CWDDSUTL, CWAASDUT, CWFXSDUT and CWDDLPUT refer to the “Batch File Utility CWDDALLU” chapter in the *Compuware Shared Services User/Reference Guide*.

CSS Language Processors (LP)

The language processors capture information about a compiler listing and store it in a source listing DDIO file. The majority of this information is gathered from the compiler listing, but optionally, SYSIN and SYSLIB are also examined. The source listing DDIO file may be in standard DDIO, shared directory, or database format.

A language processor can be run using two methods:

- preprocessor
- post-processor

You may use the preprocessor or post-processor (whichever is best for your site as needed.) For information about the pre- and post-processors, refer to the appropriate language processor chapter in the *Compuware Shared Services User/Reference Guide*. See the applicable language processor chapter for when to use the pre- or post-processor.

Language Processor Types

CSS provides language processing support for the following languages:

- COBOL
- PL/I
- Assembler
- C

The COBOL language processor reads the compiler listing produced by the COBOL compiler and writes either a compiler listing or an enhanced listing to the DDIO file. The enhanced listing merges the information normally found in the data division map (DMAP) and the condensed listing (CLIST) or procedure map (PMAP) into the source statement lines. The COBOL compiler options in effect are sorted alphabetically and appear at the beginning of the enhanced listing. The enhanced listing provides you with a condensed hardcopy of the compiler information.

The COBOL and PL/I listings may be used to provide source support using any of the following products:

- Abend-AID
- Abend-AID for CICS
- Strobe and iStrobe
- Xpediter/CICS
- Xpediter/TSO
- Xpediter/Eclipse

The Assembler listings may be used to provide source support using any of the following products:

- Abend-AID
- Strobe and iStrobe
- Xpediter/CICS
- Xpediter/TSO
- Xpediter/Eclipse

The C language processor reads the compiler listing produced by the C compiler and writes the compiler listing to the DDIO file and SYSCPRT. The source listing in DDIO can then be used to provide source support when using:

- Xpediter/TSO
- Xpediter/CICS
- Strobe and iStrobe

Security Exit Program

CSS also includes the Compuware Security Exit Program. This is an optional user-written exit for your Compuware products. It can be used in conjunction with your existing security package to secure sensitive data used by the following Compuware products:

- Abend-AID
- Abend-AID for CICS
- Xpediter/CICS
- Xpediter/TSO and Xpediter/IMS
- Xpediter/Code Coverage
- Strobe

The Compuware Security Exit enables you to control access to DDIO file members or restrict access to various commands through a user-written exit program. For the Abend-AID for CICS product, you can restrict access to some of the CWFXSDUT and CWDDALLU functions for transaction database and source listing files. For transaction databases, you can use the Security Exit Program to restrict access to individual dumps within a transaction database.

If you install a Security Exit program, it will be called by all Compuware products installed at your site. For more information on the Security Exit Program, see the "CSS Security Exit" chapter in the *Compuware Shared Services User/Reference Guide*.

Executing CSS

CSS is primarily executed at the following times:

- Abend-AID and Abend-AID for CICS
  - at compile time (when source support is used)
  - at time of abend or application program failure
  - at view time when using the Abend-AID Viewer
  - at installation and maintenance time when formatting a DDIO file.
  - when a compile listing that has been stored elsewhere is needed for Abend-AID or Abend-AID for CICS viewing with source support.
- Xpediter/TSO and Xpediter/CICS
  - at compile time (when source support is used)
  - at execution time for interactive debugging
  - at installation and maintenance time when formatting a DDIO file.
- Strobe
  - when indexing is used to analyze measurement data
  - to support Web services and SQLAF processing through CSS TP of HCI
- Compuware Workbench/Topaz Workbench
  - to support Web Services needs through the CSS TP and other HCI related facilities (SSAS and STASK)

HCI Overview

The Host Communications Interface (HCI) is a facility that provides connectivity between mainframe-based programmer productivity software and peer-node software running on other platforms in a network. These communications facilities provide a simple application programming interface so that Compuware software can be written to communicate with peer programs running on different platforms. Additionally, the communications system is flexible, simple to operate, and reliable. It has been designed to support open-ended communications protocols, and it can operate on older releases of

the MVS operating system. It allows application programs to communicate over any one of several protocols without knowing which protocol is in use at any given time.

The HCI also serves as a server activation facility that is responsible for starting and monitoring application programs that are using the communications facilities.

And finally, the HCI has been designed with no architected upper limit on the number of concurrent sessions supported, nor on throughput rates. The following figure shows how the HCI fits into the scheme of the TP application and the MVS subsystem components.

Figure 1-1. TP Subsystem, HCI and Application. HCI Position as Middleware

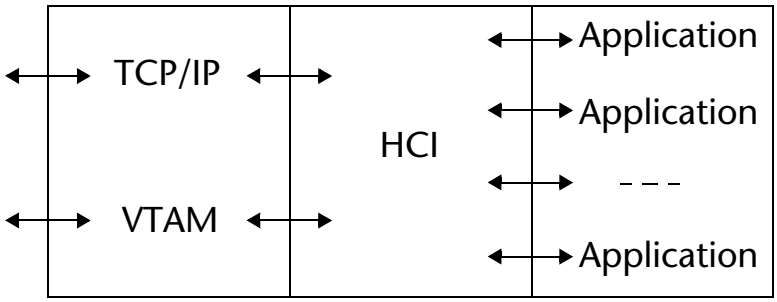
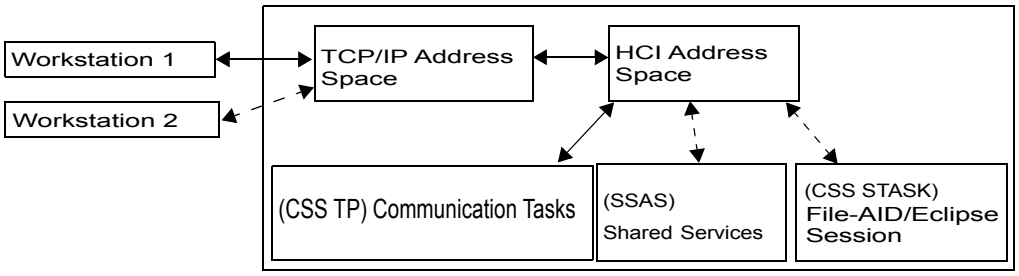


Figure 1-2. Workstation/Mainframe Connectivity



Base Services Overview

Base Services is an application framework providing batch, server, subsystem and presentation components. Functions provided by Base Services include dynamic allocation, storage management, transaction scheduling, message formatting, operator command routing, recovery and termination management, and screen, menu and message presentation in ISPF, VTAM, CICS and Web browser environments. This framework is used by Compuware Mainframe Services Controller (CMSC), Abend-AID and Abend-AID for CICS.

Beginning with Version 16.05 Base Services is part of Enterprise Common Components (ECC) installation libraries.

Special ECC 16.05 Upgrade Consideration with Abend-AID 12.04

For Abend-AID release 12.04 and prior, Base Services was installed as part of Abend-AID Common Components. However, Base Services was removed from Abend-AID release 16.05 to become part of ECC 16.05. If you are upgrading to ECC 16.05 but *not* upgrading

Abend-AID to release 16.05, then the following the procedures must be implemented to allow your existing Abend-AID installation to function correctly with ECC 16.05.

Note: You may continue to use Abend-AID release 12.04 (and below) with your prior version of CSS. However, if you want to change existing Abend-AID servers to use the ECC 16.05 load libraries then these changes will be needed:

BDCAS

In the step that executes program FDBMMPLU

- Add the 16.05 ECC SLCXAUTH as the first library on STEPLIB
- Add the 16.05 ECC SLCXLOAD as the first library on FDBDRPL
- Remove the existing SLCXLOAD from FDBDRPL

If you have steps to execute program FDBASUBS to start or stop an Abend-AID subsystem, update STEPLIB to remove the Abend-AID Common library SKAZAUTH and replace it with the 16.05 ECC SLCXAUTH library.

TDCAS

In the step that executes program FDBMMPLU

- Add the 16.05 ECC SLCXAUTH as the first library on STEPLIB
- Remove the HCI library SLHCAUTH from STEPLIB
- Add the 16.05 ECC SLCXLOAD as the first library on FDBDRPL
- Remove the existing SLCXLOAD from FDBDRPL
- Remove the HCI library SLHCLOAD from FDBDRPL

If you have steps to execute program FDBASUBS to start or stop an Abend-AID subsystem, update STEPLIB to remove the Abend-AID Common library SKAZAUTH and replace it with the 16.05 ECC SLCXAUTH library.

VIEWING SERVER

In the step that executes program FDBMMPLU

- Add the 16.05 ECC SLCXAUTH as the first library on STEPLIB
- Remove the HCI library SLHCAUTH from STEPLIB
- Add the 16.05 ECC SLCXLOAD as the first library on FDBDRPL
- Remove the existing SLCXLOAD from FDBDRPL
- Remove the HCI library SLHCLOAD from FDBDRPL

If you have steps to execute program FDBASUBS to start or stop an Abend-AID subsystem, update STEPLIB to remove the Abend-AID Common library SKAZAUTH and replace it with the 16.05 ECC SLCXAUTH library.

Chapter 2.
Preparing for ECC Installation

This chapter contains important information that can help you prepare to install the Enterprise Common Components (ECC) in the most efficient manner. Review it before beginning the installation process.

Required Components and Procedures

This section explains how to determine which parts of ECC need to be installed at your site and which chapters in this manual contain the applicable customization procedures.

Component Selection

Different Compuware products use different portions of ECC. Use Table 2-1 to determine which parts of ECC are needed at your site.

Notice that Compuware's License Management System (LMS) is required for every mainframe product, while HCI, CSS, and Base Services are components for some products.

Table 2-1. Components Utilized by Each Compuware Product

Compuware Product	CMSC	LMS	Base Services	CSS	HCI*
Abend-AID	x	x	x	x	x
Abend-AID Fault Analytics	x	x		x	
Abend-AID for CICS	x	x	x	x	x
CMSC	x		x		
File-AID/Data Solutions	x	x			
File-AID for DB2	x	x			
File-AID for IMS	x	x			
File-AID/MVS	x	x			
File-AID/RDX	x	x			
Hiperstation	x	x			x**
Strobe	x	x		x	x
Xpediter/CICS	x	x		x	
Xpediter/Code Coverage	x	x		x	
DevEnterprise	x	x		x	x
Compuware Workbench/Topaz Workbench	x	x		x	x
Xpediter/TSO and /IMS	x	x		x	
Xpediter/Xchange	x	x			

* Please refer to individual product's documentation for configuration and customization instructions.

** HCI is only required if using the Hiperstation viewer.

Required Procedures

Determining which chapters to use in this manual is actually very simple:

- Every mainframe Compuware product uses LMS. LMS runs in the CMSC address space. Therefore, every installation of a Compuware product requires, at a minimum, that the latest maintenance be applied to ECC. So regardless of which product you are installing, you must perform the applicable procedures in the following chapters:
  - Chapter 3, "Receive From Network (RFN) Installation"(if current release has not been previously installed)
  - Chapter 4, "Applying Maintenance"
  - Chapter 5, "Compuware Mainframe Services Controller (CMSC)"
  - Chapter 6, "LMS Customization"
  - Chapter 7, "HCI Customization"
- If the Compuware product you are installing requires CSS, you must also perform the procedures in Chapter 8, "CSS Customization".

The information within each chapter will direct you as to which steps apply in your situation.

ECC Packaging

The ECC product media or as part of a Receive from Network (RFN) order, includes CMSC, LMS, CSS, Base Services, and HCI, is packaged for installation using System Modification Program Extended (SMP/E). SMP/E is a method developed by IBM to install software products in the MVS environment.

SMP/E provides enhanced installation management and maintenance tracking capabilities. To facilitate SMP/E, the ECC is shipped separately from the products, via RFN or via the ECC EP physical media (upon request). The ECC software is distributed with both the base code and all cumulative, approved maintenance to-date. To install the ECC, refer to Chapter 3, "Installing Enterprise Common Components".

The installation is facilitated by the Compuware Installer consisting of a set of REXX EXECs, JCL skeletons, and panels. This facility prompts you for the installation information and then builds the necessary jobs required to perform the SMP/E installation. You can also view reporting information from the installation facility, such as parameters and execution information.

Cumulative maintenance for ECC is available for download from the Compuware GO/Frontline web site (https://go.compuware.com). It is recommended that you periodically download and apply cumulative maintenance to keep your product version current.

ECC Component Prefix and FMID

Compuware has registered the following element prefixes with IBM for the ECC components:

- **LCX** for ECC, including all of its components

FMID for ECC:

- MLCX160 — ECC base code
- NLCX160 — ECC Japanese support

Distribution and Target Libraries

The chart in Table 2-2 lists the libraries created during the ECC installation using SMP/E.

Table 2-2. Libraries Created During Installation With SMP/E

DDname	Library Type and Content	Dataset Name as Distributed
Distribution Libraries		
ALCXCNTL	ECC Distribution Sample Library	CPWR.MLCX*nnn*.ALCXCNTL
ALCXEXEC	ECC Distribution REXX Library	CPWR.MLCX*nnn*.ALCXEXEC
ALCXLOAD	ECC Distribution Load Library	CPWR.MLCX*nnn*.ALCXLOAD
ALCXMENU	ECC Distribution English Messages	CPWR.MLCX*nnn*.ALCXMENU
ALCXMJPN[1]	ECC Distribution Japanese Messages	CPWR.MLCX*nnn*.ALCXMJPN
ALCXPENU	ECC Distribution English Panels	CPWR.MLCX*nnn*.ALCXPENU
ALCXPJPN[1]	ECC Distribution Japanese Panels	CPWR.MLCX*nnn*.ALCXPJPN
ALCXSENU	ECC Distribution ISPF Skeletons	CPWR.MLCX*nnn*.ALCXSENU
ALCXSJPN	ECC Distribution Japanese ISPF Skeletons	CPWR.MLCX*nnn*.ALCXSJPN
ALCXTLIB	ECC Distribution Table Library	CPWR.MLCX*nnn*.ALCXTLIB
Target Libraries		
SLCXAUTH	ECC Target APF Authorized Load Library	CPWR.MLCX*nnn*.SLCXAUTH
SLCXCNTL	ECC Target Sample Library	CPWR.MLCX*nnn*.SLCXCNTL
SLCXEXEC	ECC Target REXX Library	CPWR.MLCX*nnn*.SLCXEXEC
SLCXLOAD	ECC Target Load Library	CPWR.MLCX*nnn*.SLCXLOAD
SLCXMENU	ECC Target English Messages	CPWR.MLCX*nnn*.SLCXMENU
SLCXMJPN[1]	ECC Target Japanese Messages	CPWR.MLCX*nnn*.SLCXMJPN
SLCXPENU	ECC Target English Panels	CPWR.MLCX*nnn*.SLCXPENU
SLCXPJPN[1]	ECC Target Japanese Panels	CPWR.MLCX*nnn*.SLCXPJPN
SLCXSENU	ECC Target ISPF Skeletons	CPWR.MLCX*nnn*.SLCXSENU
SLCXSJPN	ECC Target Japanese ISPF Skeletons	CPWR.MLCX*nnn*.SLCXSJPN
SLCXTLIB	ECC Target CSS Utilities Table Library	CPWR.MLCX*nnn*.SLCXTLIB

[1] These datasets are part of the Japanese Language support and are only installed if Japanese Language support is selected.

Note: MLCX*nnn* specifies the ECC release, where *nnn* is the ECC release number. For example, MLCX160 indicates ECC 16.05.

ECC Installation Considerations

This section describes considerations for installing and customizing ECC that applies to CMSC, LMS, CSS, Base Services, and/or HCI. Check your mainframe product installation documentation to verify if these or any of the ECC component applications have already been installed at the current release level, 16.05; look for FMID MLCX160. If they are installed, you may consider only applying maintenance. See Chapter 4, “Applying Maintenance” for more information.

Link Lists

Refer to your Compuware product documentation to determine whether the load library for the ECC components should be placed in the link list.

PDS/E Support

ECC components support Partitioned Data Set/Extended (PDS/E) libraries. Input source members for the CSS preprocessor or compiled listings for the CSS post-processor may reside in a data PDS/E library. The following are supported for PDS/E libraries:

- PDS/E object libraries are supported with DFSMS 1.1 and above.
- PDS/E data libraries are supported with DFSMS 1.0 and above.

APF Authorization

Following IBM system integrity guidelines, Compuware requires that SLCXAUTH be APF authorized. Compuware provides the SLCXAUTH library at installation time.

Following IBM system integrity guidelines, Compuware recommends that your ECC load Library (SLCXLOAD) **NOT** be APF-authorized. Depending on your MVS system installation options, libraries placed in the link list may be APF-authorized automatically.

Note: If you are using Abend-AID, the ONLY time the ECC load library should be APF-authorized is when you are using the Abend-AID load library as APF-authorized.

SLCXAUTH is also used in the Abend-AID BDCAS (Batch Dump Capture Address Space) and the Abend-AID TDCAS (Transaction Dump Capture Address Space). **For sites that are upgrading to ECC 16.05, but are not upgrading to Abend-AID 16.05, please see "Special ECC 16.05 Upgrade Consideration with Abend-AID 12.04" on page 1-7.**

LMS Installation Considerations

This section describes considerations for installing and customizing LMS.

Execution Sequence

In order to ensure availability of Compuware products with LMS-controlled access, the LMS runtime environment must be established within the CMSC address space prior to any products being initialized. For this reason, Compuware strongly recommends that you institute an MVS start-up procedure based on the sample CMSC procedure provided in SLCXCNTL. It should run automatically as part of IPL and IML processing prior to any procedures utilized by other Compuware products. It may be necessary to consult your site's MVS system programmer.

The License Management System Functions

The License Management System has three basic functions.

License Management Administration

You will create and maintain your License File using the License Administration Utility (LAU) installed with your License Management Software. Your License File is used as the source from which Compuware products will validate access for your site. When you

receive License Certificates for Compuware products, your organization's License Administrator must use the LAU to **import** the License Certificates into the License File. Additional features of the LAU assist in the maintenance of your License Files.

Depending on the requirements of your organization, you may have more than one License File. Each License File can be centrally administered from the License Administration Utility.

In addition to using the LAU to administer licenses in individual license files, an optional feature called the Centralized Licensing Facility (CLF) can be used to manage your licensing environment from a single z/OS system regardless of where your production or test LPARs reside.

Establishing the Runtime LMS Environment

You will use the License Files created and maintained by the License Administration Utility as input to a program that establishes your Compuware LMS runtime environment. , or LMZINIT when running in CLF mode, is the License Management System program that reads the License File and constructs the License Cache and License Management System subsystem against which Compuware product runtime license access requests are later made.

Important: You must re-establish your runtime License Management System environment at each IPL via the CMSC, and whenever you have an update to your License File, the update must be made available to your z/OS systems. At IPL startup, the CMSC must run LMS initialization before starting up any Compuware product.

Validating Product Access Requests During Product Use

Your access to your Compuware products will be validated when you use your Compuware products. The Compuware product will make a request of the License Management System to determine if your site has a valid License Certificate for the product release. The given Compuware product will access the LMS subsystem established, request License File information and act upon the information. Any abnormal License Management event detected will be reported to the product's user and may optionally be reported by e-mail to your organization's License Administrator. The disposition and frequency of the reporting of these events can be controlled by the customer using the "Message Suppression and Distribution" feature of LMS. Optionally, these events may also be recorded in SMF Logging.

Note: LMS can read more than one License File if you are an organization that administers Compuware product access for more than one Compuware customer.

CSS Installation Considerations

This section describes considerations for installing Compuware Shared Services which is part of the Enterprise Common Components (ECC) libraries.

CICS Load Library Definition

The ECC load library (SLCXLOAD) must be included with Compuware product libraries in the CICS startup JCL or RDO definitions for the following product:

- Xpediter/CICS (all supported releases)

TSO Logon PROCs

If the ECC load library (SLCXLOAD) is not in the link list and dynamic library allocation is not used, the ECC load library must be placed before all Compuware product load

libraries in the STEPLIB or ISPLLIB concatenations in the TSO login PROC. The ECC message and panel libraries must be placed before all Compuware product message and panel libraries in the ISPMLIB and ISPPLIB concatenations in the TSO login PROC.

Sharing CSS DDIO Files With Multiple CPUs

If you are sharing CSS DDIO files among multiple CPUs, the following qnames must be changed from LOCAL to GLOBAL enqueues.

- ABENDAID
- ABENDDAM
- ABENDSMF
- CWCATALG

Note: These qnames are needed for **all** products. Please refer to the individual Compuware product documents with which CSS was distributed for other product requirements.

For IBM's GRS (Global Resource Serialization), these qnames must be added to the inclusion RNL. This RNL is defined in the GRSRNLxx member of SYS1.PARMLIB for MVS/XA and later. For pre-MVS/XA systems, these qnames are maintained in the ISGSIRNL entry point of ISGGRNLO. Please consult the IBM product documentation for further details.

The ECC sample library (SLCXCNTL) contains an example of how qnames should be coded in the GRSRNLxx member of SYS1.PARMLIB. See "CSS Sample Library Members" on page 2-9 for more information.

For resource serialization packages other than GRS (for example MIM), consult the product documentation for changing LOCAL to GLOBAL enqueues.

CSS Language Processor (LP)

The language processor captures information about a compiler listing and stores it in a source listing DDIO file. The majority of this information is gathered from the compiler listing, but optionally, SYSIN and SYSLIB are also examined. The source listing DDIO file may be in standard DDIO, shared directory, or database format. Releases prior to Xpediter/TSO 7.3, Xpediter/CICS 7.6, and Abend-AID for CICS 4.5 do not support source listing shared directories. Instead, individual source databases may be specified for lower levels.

The language processor can be run using two methods:

- pre-processor
- post-processor

You may use the pre-processor or post-processor (whichever is best for your site as needed). This section discusses the benefits of the preprocessor and the post-processor and describes when to use them.

Pre-processor

Some information about a program module is not always available from the compiler listing. The pre-processor, in addition to gathering information from the compiler listing, gathers information from SYSIN and SYSLIB. The preprocessor provides additional functionality for Xpediter/CICS and Xpediter/TSO, and is required for PL/I programs that use the %NOPRINT statement.

When setting up the language processors, Compuware recommends that you use the pre-processor instead of the post-processor. The post-processor should only be used in situations where compiled source listings have been stored.

Pre-processor Steps

1. Determines the proper compiler options required to process the listing file.
2. Automatically invokes the compiler or assembler to compile or assemble your source program.
3. Writes the listing to the DDIO file. An enhanced listing for COBOL programs can be produced, if desired. PL/I and Assembler listings are written to SYSPRINT. C listings are written to SYSCPRT.

Pre-processor Benefits

- **Better handling of compiler errors.** By using the preprocessor, you can avoid the potential problem of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 8, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 8. The preprocessor can process the compiler errors more effectively than the post-processor. Compuware recommends that you use the preprocessor rather than the post-processor.
- **Capturing suppressed source code.** When any of the following parameters are used, sections of source code can be suppressed from the compiler listing:

For COBOL	COPY SUPPRESS
For PL/I	%NOPRINT
For Assembler	PRINT OFF or PRINT NOGEN
For C	NOSHOWINC and NOEXPMAC

  The preprocessor captures this information from SYSIN and SYSLIB, or by forcing on more revealing compiler options, then altering the listing to emulate the options you requested.
- **Automated compiler options.** The post-processor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.
- **Simplified JCL.** While the post-processor requires that the user add a step after the compile step, the preprocessor requires only that minor modifications be made to your existing compile step.

Files Dynamically Allocated

The following files are dynamically allocated by the Compuware preprocessor. The attributes can be overridden by specifying a DD in the JCL.

Table 2-3. Files Dynamically Allocated by the Preprocessor

DDname	Attributes
CWPWRK*n* (0-6)	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
SORTWK01	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
TEMPLIN	BLKSIZE=3200 SPACE(TRK,(200,100)) UNIT=SYSDA
CWPERRM	SYSOUT=*
SYSPRINT	SYSOUT=*
SYSOUT	SYSOUT=*
CWPPRTI (PL/I)	BLKSIZE=19000 SPACE(TRK,(100,200)) UNIT=SYSDA
CWPPRTI (all other languages)	BLKSIZE=16093 SPACE(TRK,(100,200)) UNIT=SYSDA DISP=MOD
CWPPDS2	SPACE (TRK,(100,80,80)) UNIT=SYSDA

CWPPRTO is dynamically allocated as a work file (for the preprocessor). SYSPRINT is where the listing goes.

Post-processor

The postprocessor is executed as a single step and it can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate the use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in the appropriate language processor chapters in the *Compuware Shared Services User/Reference Guide*.In order for the listings to populate the DDIO, they must be compiled using the required options specified in the appropriate language processor chapter. Note that these are usually the default options at many shops.

If the programs contain suppressed source code, certain additional control statements must be added to the language processor. Please view the Language Processor chapters in the *Compuware Shared Services User/Reference Guide* for the programming language desired for more details.Please view the Language Processor chapters for the programming language desired for more details.

If the programs are compiled with OFFSET as well as OPT(imize), certain additional control statements must be added to the language processor. Please view the Language Processor chapters in the *Compuware Shared Services User/Reference Guide* for the programming language desired for more details.Please view the Language Processor chapters for the programming language desired for more details.

Postprocessor Benefits

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.
- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Language Processor DD Statements

Table 2-6 lists the DD statements for the Assembler, COBOL, PL/I, and C language processors. Not every DD Name applies to every language processor.

Table 2-4. Language Processor DD Statements.

DDname	Purpose
CWPWRK (0-6)	LP work files
CWPPRMO	LP parameter input dataset
CWPDDIO	Target source listing DDIO file
CWPPRTI	Compiler listing input to postprocessor
CWPPRTO	Compiler listing output from postprocessor
CWPLOAD	Option LOAD or OBJECT module input to postprocessor
CWPDECK	Option DECK module input to postprocessor
CWPERRM	LP Error Messages

Table 2-4. Language Processor DD Statements.

DDname	Purpose
CWPWBNV	Workbench navigation information

Note: CWPLOAD and CWPDECK correspond to the SYSLIN and SYSPUNCH compiler outputs. If you specify compiler option OBJECT, you should use CWPLOAD in your postprocessor JCL. If you specify compiler option DECK, you should use CWPDECK in your postprocessor JCL. Do not feed the SYSLIN output into CWPDECK or the SYSPUNCH output into CWPLOAD. The updated OBJECT contained in CWPLOAD and CWPDECK should be used in the program's linkedit step. The preprocessor does not use CWPLOAD and CWPDECK.

Writing a Listing to a DDIO File

Several options exist to place listings in a DDIO file. Each option requires that some changes be made to your JCL. The type of changes needed depend on whether you are using the preprocessor or the post-processor, and the type of debugging session. Refer to "CSS Sample Library Members" for a list of the sample JCL that can be used for the various options.

Testing and Debugging Programs

The actual method used for handling testing and debugging programs depends on whether you are preprocessing or postprocessing:

- **Preprocessing** — Run the preprocessor and link your program.
- **Postprocessing** — Compile your program, run the post-processor, and then link your program.

With either of these methods, the language processor places a listing in the DDIO file whenever a program is compiled.

Debugging Production Programs

If the original compiler listing is available in machine-readable format, you can retrieve the listing and use it as input to the post-processor in order to reprocess the listing and recreate it in the DDIO file. The post-processor can process listings stored by most products.

Note: If the listing you use as input to the post-processor contains any other information (such as translator or link edit output), in addition to the compiler information, you may not obtain the desired results.

CSS Sample Library Members

The following list describes the CSS members of the ECC sample JCL library that you may work with during installation (depending on your installation options). Use these members only as needed if directed to do so by a specific Compuware product installation guide. These members are distributed and updated via SMP/E. The members will be located in the following files (showing initial default HLQ) after the SMP/E Apply and Accept steps are complete.

- CPWR.MLCX*nnn*.SLCXCNTL (target library)
- CPWR.MLCX*nnn*.ALCXCNTL (distribution library)

where *nnn* indicates the release number, for example MLCX160.

To print the CSS sample library members, submit the JCL contained in library member CXPRINT after the distributed media has been unloaded.

CWASSECD Security Exit program DSECT.

CWASSECU Sample Security Exit program.

CWCMTRHE Sample customized horizontal translation table for mixed-case English.

CWCMTRHT Sample customized horizontal translation table for uppercase English.

CWCMTRHU Sample customized horizontal translation table for mixed-case English with Euro character.

CWCMTRVE Sample customized vertical translation table for mixed-case English.

CWCMTRVT Sample customized vertical translation table for uppercase English.

CWCMTRVU Sample customized vertical translation table for mixed-case English with Euro character.

Note: The following five members are specific for Abend-AID. They allow browse access to diagnostic reports and source listings directly through ROSCOE or print access via a job submitted from ROSCOE to run in batch. Refer to the members for additional information.

CWROSBAT ROSCOE sample members.

CWROSBRW ROSCOE sample members.

CWROSCOE ROSCOE sample members.

CWROSDIR ROSCOE sample members.

CWROSPDF ROSCOE sample members.

CWUTCLSE Sample CLIST to invoke CSS online utilities with English messages and panels.

CWUTCLSJ Sample CLIST to invoke CSS online utilities with Japanese messages and panels.

CWUTREXE Sample REXX EXEC to invoke CSS online utilities with English messages and panels.

CWUTREXJ Sample REXX EXEC to invoke CSS online utilities with Japanese messages and panels.

CWVFCLPT CLIST to invoke the language processor debugging aid.

Note: The debugging aid is a diagnostic tool to be used only under the supervision of Compuware Customer Support in the event of a problem related to the language processor.

CWVFRXPT Sample REXX EXEC to invoke the Compuware language processor debugging aid.

Note: The debugging aid is a diagnostic tool to be used only under the supervision of Compuware Customer Support.

CXAADIRX Abend-AID CWAASDUT DIRX sample JCL.

CXAAEXPO Abend-AID CWAASDUT EXPORT command sample JCL.

CXAAIMPO Abend-AID CWAASDUT IMPORT command sample JCL.

CXAAMOVE Abend-AID CWAASDUT MOVE command sample JCL

CXALLBAA JCL for creating sequential Abend-AID database files.

CXALLBLP JCL for creating sequential source listing database files.

CXALLBSD JCL for creating Abend-AID for CICS sequential transaction databases.

CXALLDAA JCL for creating Abend-AID shared directories.

CXALLDLP JCL for creating Source Listing Shared Directories.

CXALLDS JCL to allocate a sequential DDIO file.

CXALLMC JCL for creating Abend-AID for CICS shared directories.

CXALLVAA JCL for creating VSAM Abend-AID database files.

CXALLVLP JCL for creating VSAM Source Listing Database files.

CXALLVS JCL to allocate a standard VSAM Abend-AID report or source listing DDIO file.

CXALLVSD JCL for creating Abend-AID for CICS VSAM transaction databases.

CXASM JCL for running the Assembler language post-processor.

CXASMPRE JCL for running the Assembler language preprocessor.

CXC Sample JCL for running the C post-processor.

CXCAPTUR Sample JCL for creating a tape from the output of the CSS Problem Documentation Utility.

CXCFGEXT Sample JCL to extract a source file from the configuration module in an ECC load library.

CXCFGINI Sample source configuration file.

CXCFGSET Sample JCL to build a configuration module in an ECC load library, from a source configuration file you create.

CXCIREG Sample JCL to register the Contact Information dataset.

CXCIVSAM Sample JCL to create the Contact Information dataset.

CXCOBPRE JCL for running the COBOL language preprocessor.

CXCOB1 JCL for the step to be added after the compile step in your current COBOL compile and link edit JCL. This is used in order to process the compiler listing through the Compuware language post-processor.

CXCOB2 JCL to process COBOL compiler listings stored in machine-readable format.

CXCOB99 Sample JCL for running the COBOL post-processor.

CXCPRE Sample JCL for running the C preprocessor.

CXC2 Sample JCL to postprocess C compiler listings stored in machine-readable format.

CXDCRADB JCL to allocate and CREATE a multi-volume Abend-AID report database.

CXDCRASD JCL to allocate and CREATE an Abend-AID shared directory and a multi-volume report database.

CXDCRLDB JCL to allocate and CREATE a multi-volume source listing database.

CXDCRLSD JCL to allocate and CREATE a source shared directory and a multi-volume Source Listing database.

CXDCRTDB JCL to allocate and CREATE a multi-volume transaction database.

CXDCRTSD JCL to allocate and CREATE an Abend-AID for CICS shared directory and a multi-volume transaction database.

CXDDSUTL	Sample JCL for running the CWDDSUTL batch file utility.
CXDDUNLP	Sample JCL to execute the CWDDUNLP source extraction utility.
CXDFMAA	JCL to allocate and FORMAT a standard multi-volume Abend-AID report DDIO file.
CXDFMSL	JCL to allocate and FORMAT a standard multi-volume source listing DDIO file.
CXEXPORT	Sample JCL for running the CWDDSUTL EXPORT command.
CXFLAG	Sample JCL for the CWDDSUTL utility FLAG command.
CXFMTAA	Sample JCL and CWDDSUTL control statement for formatting Abend-AID report files.
CXFMTSL	Sample JCL and CWDDSUTL control statement for formatting a source listing file.
CXCFDIRX	Abend-AID for CICS CWFXSDUT DIRX sample JCL.
CXCFEXPO	Abend-AID for CICS CWFXSDUT EXPORT command sample JCL.
CXCFEXTL	Sample JCL to export a Abend-AID for CICS transaction **and** a source listing to a single tape.
CXCFIMPO	Abend-AID for CICS CWFXSDUT IMPORT command sample JCL.
CXCFMOVE	Abend-AID for CICS CWFXSDUT MOVE command sample JCL.
CXIMPORT	Sample JCL for running the CWDDSUTL IMPORT command.
CXJCLSEC	JCL to assemble and link edit the Security Exit program.
CXJCLTRT	JCL to assemble and link edit the custom translation tables.
CXLPASM	Sample Assembler language processor options.
CXLPC	Sample C language processor options.
CXLPCOBB	Sample COBOL language processor options (batch programs).
CXLPCOBC	Sample COBOL language processor options (CICS programs).
CXLPDIRX	LP source CWDDLPUT DIRX sample JCL.
CXLPEXPO	LP source CWDDLPUT EXPORT command sample JCL.
CXLPIMPO	LP source CWDDLPUT IMPORT command sample JCL.
CXLPMOVE	LP source CWDDLPUT MOVE command sample JCL.
CXLPPLI	Sample PL/I language processor options.
CXLPTFS	Sample JCL to run SMPE LIST on ECC Target/Distribution zones.
CXLZAPS	Sample JCL to list the PTFs and APARs on an ECC load library
CXMODMAP	Sample JCL member to map the CSECTs in a load module.
CXOTFPRE	Sample CLIST stub for preprocessing exit for OTF
CXOTFPST	Sample CLIST stub for postprocessing exit for OTF
CXPDDSU	Sample JCL to print a listing or report from a DDIO file.
CXPLI	JCL for running the PL/I language post-processor.
CXPLIPRE	JCL for running the PL/I language preprocessor.
CXPLI2	JCL to postprocess PL/I compiler listings stored in machine-readable format.

CXPRCTL	IEBPTPCH control statements for job CXPRINT.
CXPRINT	JCL to print the entire ECC sample library SLCXCNTL.
CXRELS	PTF/APAR tracking module. **Do not modify this member.**
CXR00003	Sample Tutorial main menu.
CXSPRE01	Sample Skeleton for on-the-fly preprocessing.
CXSPST01	Sample Skeleton for on-the-fly postprocessing.
CXTUTOR	Alternate Sample Tutorial main panel.
CXUECEXT	Sample CLIST for the Compile Facility JOB card exit
CXUECIN2	REXX EXEC used to invoke the Compile Information installation screens
DDIODAYS	REXX utility to delete DDIO file members by age.
DDIODAYJ	Sample JCL for running the DDIODAYS REXX utility. This utility deletes DDIO file members by age.
FDXTRN01	File-AID DB2 Interface CLIST.
GRSRNLXX	Sample GRS qname inclusion RNL.
P@CPRE01	Sample panel for preprocessing exit for on the fly.
P@CPST01	Sample panel for postprocessing exit for on the fly.
P@HPRE01	Sample Help panel for preprocessing exit for on-the-fly processing.
P@HPST01	Sample Help panel for postprocessing exit for on-the-fly processing.
ATSOECLP	Sample TSO logon procedure for Xpediter/TSO via Compuware Mainframe Eclipse.
CSPFINIT	Sample JCL to create and initialize the Compuware Shared Profile dataset.
CXCFGBLD	Sample JCL to create the Compuware Mainframe Eclipse configuration file dataset.
CXTPDEF	Sample HCI configuration parameters to be inserted into an existing HCI configuration.
HCIPROCS	Sample JCL to create HCI and CSS TP procedures.
HCIJOURN	Sample JCL to create the HCI journals.
HCITEST	Sample JCL to test HCI connectivity.
HCI00	Sample HCI and CSSTP PARMLIB member.

HCI Installation Considerations

This section describes considerations for installing the Host Communications Interface which is a requirement for several Compuware products.

Security Settings

HCI can switch ACID (UserID) while processing certain requests. Some security package settings may then prevent the HCI from writing to JES sysout after this is done. Installation with such strict security settings need to allow the HCI task access to the JESSPOOL class.

Installations that have such settings that limit any or all user access to output from other users can do so by setting the appropriate profiles for the security class JESSPOOL.

Profiles for JESSPOOL are in the format:

```
Local-nodename.userid.jobname.jobid.dsnumber.name
```

Using combinations of userIDs and wildcard characters, output viewing can be limited on a userid or groupid basis as the installation chooses.

Please consult your installation's appropriate security software administration documentation for additional information. For RACF installations, controlling access to output via the JESSPOOL class is documented in IBM's *z/OS Security Server RACF Security Administrator's Guide*, section entitled, "Controlling Access to Spool Data".

Base Services Installation Considerations

This section describes considerations for installing Base Services which is a requirement for CMSC, Abend-AID, and Abend-AID for CICS.

For sites that are upgrading to ECC 16.05, but are not upgrading to Abend-AID 16.05, please see "Special ECC 16.05 Upgrade Consideration with Abend-AID 12.04" on page 1-7.

TSO Logon PROCs

Add the Base Services library (CPWR.MLCX*nnn*.SLCXLOAD) to the ISPLLIB concatenations. It may also be added to your TSO logon PROC or added to the link list. For more information, see "Step 5. Activate Compuware ISPF Dialogs (Required)" on page 8-15.

Chapter 3.
Receive From Network (RFN) Installation

Installation Methodology

Compuware's primary product delivery and installation method is Receive From Network (RFN). The RFN process enables you to receive the product directly to your z/OS environment and install it using System Modification Program Extended (SMP/E).

After you select a licensed product from Compuware's online ordering system, you will receive an e-mail message containing sample JCL for creating a Compuware Installer dialog PDS.

Documentation for using the Compuware Installer to install the Enterprise Common Components (ECC) is contained in the *Compuware Installer Mainframe Product SMP/E Installation Guide* document attached to the order e-mail.

If you ordered the ECC EP Physical media, you can obtain a copy of this document from Compuware GO (https://go.compuware.com) in the ECC documentation section. The document also contains information for using the EP physical media to SMP/E install Enterprise Common Components.

Compuware recommends to install Enterprise Common Components only from one delivery method (RFN or EP media).

Notes:

1. Abend-AID Release 16.05 or newer, Compuware Program Analyzer Release 16.05 or newer, and DevEnterprise Release 16.5 or newer require ECC Release 16.05 or newer.
2. Enterprise Common Components (ECC) Release 16.05 or newer cannot be in the same zones as Abend-AID 12.4 and prior releases, Compuware Program Analyzer Release 5.3, or DevEnterprise 5.3 and prior releases.

Chapter 4.
Applying Maintenance

Maintenance Delivery Options

All maintenance for each supported Compuware product release is provided on Compuware GO (https://go.compuware.com). Select **Enterprise Common Components** from the **Product Support: My Product** selection menu, the **Enterprise Common Components** web page appears. Locate and select the **Fixes/Downloads** link and select the desired product and release for which to apply maintenance.

Follow the instructions on the web site for acquiring and applying cumulative maintenance.

Optionally, you may apply maintenance by getting a current copy of the ECC EP physical media and using it to generate the JCL for applying maintenance through SMP/E. Refer to the *Compuware Installer — Mainframe Products SMP/E Installation Guide* for additional details on applying maintenance from Compuware EP media.

CAUTION:

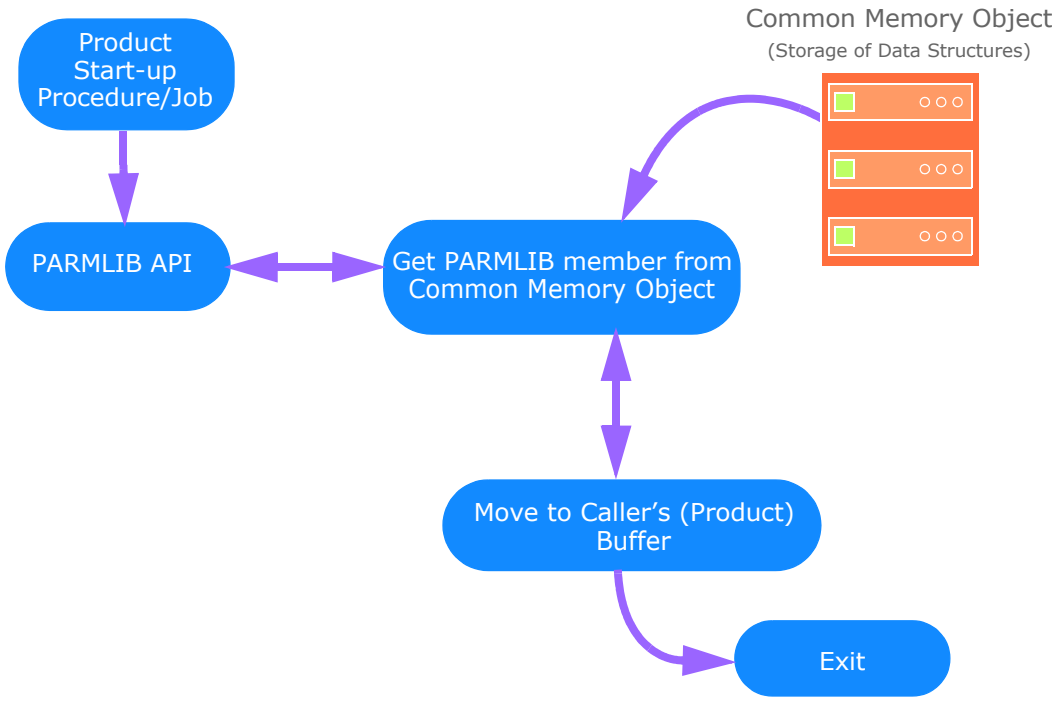
Products using address spaces or server regions where ECC libraries are allocated must be shut down while maintenance is being applied to their ECC libraries and restarted after the maintenance process is complete. This applies especially to Abend-AID for CICS; HCI regions supporting the Compuware Workbench[1], Topaz Workbench, Strobe and DevEnterprise; and Xpediter/CICS users.

1. The Compuware Workbench has been replaced by the Topaz Workbench. The Topaz Workbench provides all of the functionality of the previous Compuware Workbench as well as any additional chargeable functionality.

Chapter 5.
Compuware Mainframe Services Controller (CMSC)

The Compuware Mainframe Services Controller (CMSC) address space is a centralized facility providing license management and common Parameter Library Services for Compuware mainframe products. The CMSC also has the ability to automatically start HCI address spaces. HCI address spaces started by the CMSC will automatically restart in the event of a failure.

Figure 5-1. Compuware Mainframe Product Parameter Retrieval flowchart



Running the CMSC

IMPORTANT:

If you are installing ECC Release 16.05 for the first time, you *must* customize License Management System (see Chapter 6, "LMS Customization") before running CMSC.

If the CMSC is to be used to invoke HCI as a started task, then HCI needs customized prior to running CMSC (see Chapter 7, "HCI Customization").

You can execute the CMSC from JCL as a batch job or from a procedure as a started task. **Compuware recommends execution as a started task**. The ECC installation sample library (SLCXCNTL) contains the relevant members.

- Member CMSC includes sample JCL that can be executed in a batch job, or can be executed directly as a started task. Follow the instructions in the member, modify the JCL to your site's requirements then submit.

The format of the PARM parameter of the EXEC JCL statement is:

`PARM='name,CMSC,ssss':`

- ***name*** is the name of the CMSC. This can be any unique name of your choosing up to 8 characters in length. The only restrictions are that it must begin with a letter or national character, and contain letters, digits or national characters. **This parameter is not related in any way to the CMSC_ID start-up parameter.**
- **CMSC** identifies the address space being initialized by this JCL as a CMSC.
- ***ssss*** specifies the 1 to 4 character suffix for the CMSC PARMLIB member. This parameter will be concatenated with the prefix "CMSC" to construct the name of the PARMLIB member to be used for the CMSC start up parameters. The suffix must follow the rules for PDS members.

Step 1: Configure the CMSC Start-Up Parameters

CMSC00 is the default CMSC PARMLIB member. In the CMSC00 PARMLIB member, modify the CMSC parameters to your site's requirements before starting the CMSC.

CMSC_ID=*cmsc_id*

Description: Specifies the name of the CMSC. This parameter must be a 1 to 4 character string. The string must consist of the letters (A-Z), digits (0-9), or national symbols (@#$). It must begin with a letter or national symbol.

`CMSC_ID=CMSC` must be specified for the primary CMSC.

Use the CMSC_ID start-up parameter when your site requires multiple instances of CMSC. For example `CMSC_ID=TEST` identifies a secondary CMSC, with the CMSC_ID of TEST, dedicated to testing purposes.

Default: none

Required: yes

CES_PORT=*ces_port_number*

Description: A decimal number from 1024–65535 that specifies the port number for the Compuware Enterprise Server (CES) to which the CMSC is to send REST data.

Default: none

Required: no

CES_HOST=*ces_host_name*

Description: Up to 70 characters specifying the hostname or IP address of the Compuware Enterprise Server (CES) to which the CMSC is to send REST data.

Note: To enable CES support the CES_* parameters are required.

Default: none

Required: no

CLF=NO|YES

Description: CLF statement is used to specify whether LMS should run in Central License Facility mode.

Default: NO

YES enables Centralized License Facility (CLF) mode.

NO disables Centralized License Facility (CLF) Mode.

Required: no

LMS_CLIENT=NO|YES

Description: LMS_CLIENT statement is used to indicate whether the License Management System (LMS) client should run in the CMSC address space.

Default: YES

YES indicates that the License Management System (LMS) client should run in the CMSC. If not running in CLF mode, this option indicates that — the LMS program that reads the License File and constructs the License File cache against which runtime product access requests are made — is to run in the CMSC address space.

NO indicates that the License Management System (LMS) client should not run in the CMSC address space.

Required: no

LMS_SERVER=NO|YES

Description: LMS_SERVER statement is used to specify whether the License Management System (LMS) Central License Facility (CLF) server should run in the CMSC address space.

Default: NO

YES indicates that the LMS CLF server should run in the CMSC.

NO indicates that the LMS CLF server should not run in the CMSC.

Required: no

HCI_PROC=*procedure_name*

Description: Specifies the name of the HCI start-up procedure.

Default: none

procedure_name is the procedure name of the HCI start-up procedure.

Required: no

HCI_PROC*n*=*procedure_name*

Description: Specifies the names of additional HCI start-up procedures. Up to 8 HCI address spaces can be started and controlled by the CMSC.

Default: none

procedure_name is the name of the HCI start-up procedure for HCI_PROC*n*, where *n* is an integer from 1 to 8.

Required: no

ZIIP_ENABLEMENT=NO|YES

Description: Enables the Compuware zIIP service which other products use with various functionality.

Default: NO

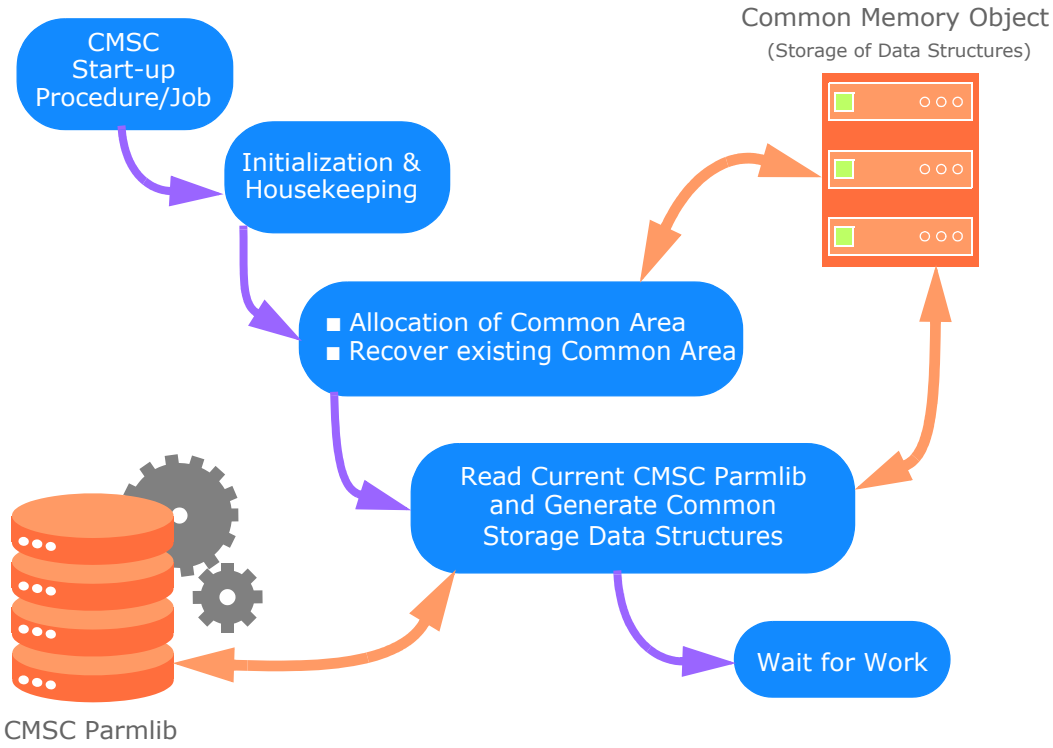
YES indicates that the Compuware Shared Services zIIP Enablement service is to be started.

NO indicates that the Compuware Shared Services zIIP Enablement service is disabled. This is the default.

Required: no

Note: If none of the License Management-related CMSC parameters (CLF, LMS_CLIENT, and LMS_SERVER) are specified, the default action is to run License Management in non-CLF mode and run in the CMSC using the parameters specified in the LMCL CMSC parameter (default LMCL00 if the LMCL parameter is not specified).

CMSC Start-up Flowchart



Overriding Default CMSC and Parameter Suffix Values

Specify the following DDs a given products JCL or started procedure in order to override the default CMSC and it's default PARMLIB member suffix values.

//CWSCssss DD DUMMY	Used to override the default CMSC in which the HCI will retrieve parameters. Where ssss is the four-character CMSC identifier specified on the CMSC PARM=.
//PMaaNNNN DD DUMMY	Optionally used to override the default PARMLIB member specified in the CMSC configuration. Where aa is the two-character product code and NNNN is the PARMLIB member suffix.

CMSC Commands

Use the z/OS MODIFY command for the following commands. Note: The verb MODIFY can be abbreviated, F.

PARMLIB Commands

Use the following commands to control the Parameter Library Service of the CMSC:

Refreshing All Parameter Members:

```
F cmscname,PARMLIB REFRESH
```

Refreshing a Single Parameter Member:

```
F cmscname,PARMLIB REFRESH  member_name
```

Displaying the Current Parameter Libraries:

```
F cmscname,PARMLIB DISPLAY
```

Stopping the Parameter Library Service:

```
F cmscname,PARMLIB STOP
```

Note: The PARMLIB STOP command will stop the Compuware Parameter Library Service. All requests for Parameter Library Services will result in a nonzero return code. It will not shut down the CMSC. In order to restart the Parameter Library Service the CMSC will have to be restarted. **This command should only be used in emergency situations under direction of Compuware Customer Support**.

LMSHUT

Shutting Down the Compuware License Management Client or Server Task in the CMSC:

```
F cmscname,LMSHUT CLIENT|SERVER
```

LMRESTRT

Restarting the Compuware License Management Client or Server Task in the CMSC:

```
F cmscname,LMRESTRT CLIENT|SERVER
```

STARTHCI

Restarting an HCI Address Space Controlled by the CMSC:

```
F cmscname,STARTHCI hci_procname
```

Note: This command only applies to HCI address spaces controlled by the CMSC using the HCI_PROC CMSC startup parameter.

STOPHCI

Shutting Down an HCI Address Space Controlled by the CMSC:

```
F cmscname,STOPHCI hci_procname
```

Note: This command only applies to the HCI address spaces controlled by the CMSC using the HCI_PROC CMSC startup parameter.

SHUTDOWN

Stopping the CMSC:

```
F cmscname, SHUTDOWN nnnn|IMMEDIATE
```

Note: *nnnn* — A timer value that specifies the number of seconds to elapse before CMSC terminates. The default value is 60 seconds.

IMMEDIATE — A timer value that specifies that the CMSC is to stop immediately.

zIIP Commands

Start zIIP service:

```
F cmscname,ZIIP INIT
```

Stop zIIP service:

```
F cmscname,ZIIP DELETE
```

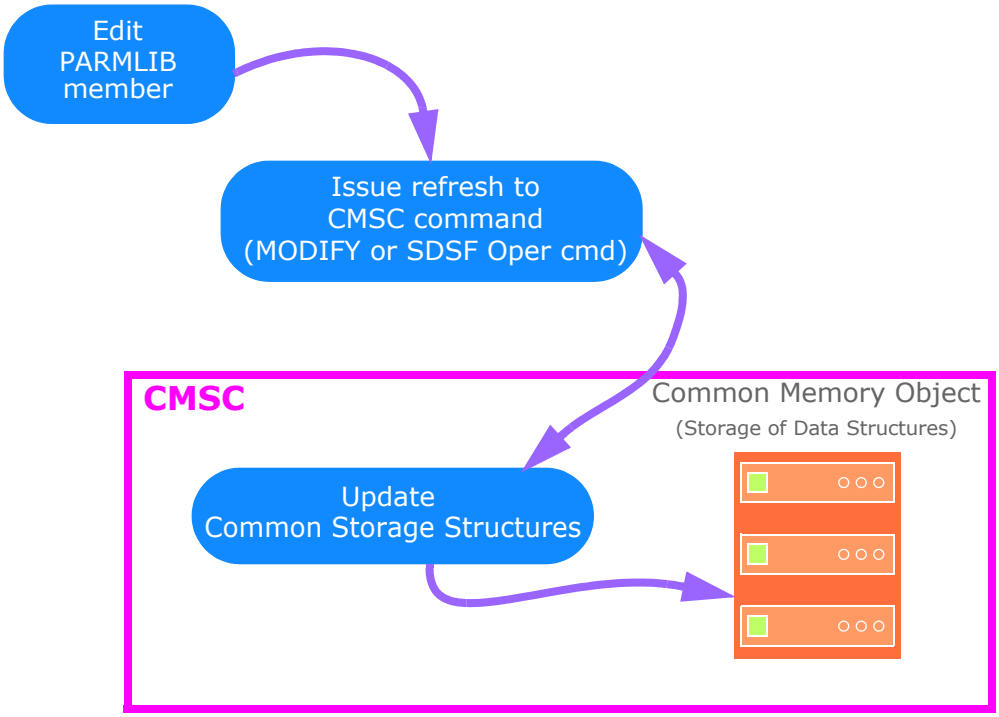
List zIIP service blocks:

```
F cmscname,ZIIP LIST
```

Continue with ECC Customization

Continue with the procedures in **Chapter 6, "LMS Customization"** to customize the License Management System (LMS).

Changing a PARMLIB Member



Configuring Compuware Mainframe Products

Once the SMP/E installation of your Compuware mainframe products has completed, this section can be used to configure and customize them.

The Compuware Mainframe Services Controller (CMSC) provides storage and retrieval functions in relation to your Compuware mainframe product's configuration parameters. **Compuware mainframe products are configured using PARMLIB members. Using this method may be fundamentally different from the way you have configured your Compuware mainframe products in previous releases.** Parameters are modified by editing human-readable PARMLIB members, then telling CMSC to store those modified PARMLIB members into the Common Memory Object by issuing a refresh (MODIFY) command. Compuware mainframe products receive parameter values stored in the Common Memory Object.

Note: The Common Memory Object containing your mainframe product parameters can be accessed regardless of whether the CMSC is active. When the CMSC is initialized all PARMLIB members are be loaded into a Common Memory Object.

PARMLIB Overview

Compuware mainframe products use parameter libraries, or PARMLIBs for setting their configuration and customizations values. Compuware recommends storing your site's PARMLIB members in a common library. A copy of your product's parameter files must reside in the **//CWPARM concatenation of the Compuware Mainframe Services Controller (CMSC)**.

The CWPARM is the DD from which the CMSC retrieves the parameters. This DD is a concatenated list of all datasets containing the PARMLIB members for each of the Compuware mainframe products. The CMSC will only read members into storage that contain the product prefixes specified in Table 5-1. Sample parameter files provided in SLCXCNTL may be copied to the desired CWPARM dataset and then modified to site-specific requirements.

CMSC start-up parameters (Table 5-1) specify the default 1 to 4 character suffix for all the Compuware product PARMLIB members. If the parameter value for a given product is omitted (null or blank), the default suffix will default to 00.

The product parameter is specified followed by the equal sign, and then the 1 to 4 character suffix (for example: `LMCL=01` points to PARMLIB member LMCL01).

Table 5-1. Compuware Product PARMLIB Member Parameters

Product PARMLIB Member Parameters	Product
AABD	Abend-AID BDCAS
AADC	Abend-AID DCAS
AAFA	Abend-AID Fault Analytics
AATD	Abend-AID TDCAS
AAVW	Abend-AID Viewer
CMSC	Compuware Mainframe Service Controller
FACM	File-AID Common Component
FADA	File-AID/Data Solutions
FADE	File-AID DB2 Environment Information
FAMV	File-AID/MVS
FAFR	File-AID/RDX
FAFD	File-AID *for DB2*
FAIE	File-AID IMS Environment Information
FAIX	File-AID *for IMS*
HCI	Host Communications Interface
HSGR	Hiperstation
LMCL	License Management Client ()
LMSV	License Management Server (LMZINIT)
STR	Strobe

General Guidelines for PARMLIB Members

Parameters used by your Compuware mainframe product or component are read from the common PARMLIB dataset. Edit the sample parameters to your site's requirements.

Table 5-2. Guidelines for PARMLIB Members

PARM Name	Values	Samples
Data columns	1 to 71	N/A
Symbol separator	Underscore	"A_LONG_SYMBOL"
Continuation	check first for any character in Column 72 otherwise, check for "," is the last non-blank	1. any non-blank 2. ","
Comments	"*" in column 1 between columns 1-71, begin with "/* and end with "*/"	* is a comment /* is a comment */

- If the PARMLIB member includes multiple groups of parameters, for example for the definition of multiple DB2 Subsystem, then only one occurrence of each of the parameters within each group is allowed.
- An asterisk '*' in column 1 causes the entire statement to be a comment.
- Line level comments are supported using the '/*' to start a comment and '*/' to end the comment. Embedded comments are supported.

Using System Symbolics

The CMSC resolves system symbolics as product parameters are saved into storage, potentially allowing a single parameter member to be used across multiple LPARs. These symbols are defined by your installation in IEASYMxx. Issue the following display command to display the current symbols.

/DISPLAY SYMBOLS

PARMLIB Member Naming Convention

PPPPnnnn

PPPP is the product prefix (see Table 5-1).

nnnn is the 1 to 4-character PARMLIB suffix, for example 00.

The PARMLIB member name for each product or product component must start with the product prefix, for example FACM for File-AID Common Components. The 1 to 4-character suffixes can be used to replace the default name (FACM00). If the parameter for a given product is omitted, the default suffix will be 00.

Changing the Default PARMLIB Member

In the CMSC start-up parameters, specify the product parameter, followed by the equal sign, and the 1 to 4-character suffix (for example: FACM=01, which points to PARMLIB member FACM01. If you changed your PARMLIB member name from the default FACM00 to, for example FACM0005, update your CMSC PARMLIB member to point to the new PARMLIB member, FACM=0005.

General Guidelines for PARMLIB Dataset:

- The dataset can be blocked.
- The dataset can have multiple extents.
- The dataset must be on a single volume.
- The CMCS must have READ access.

PARMLIB Product Configuration

Step 1. Copy Sample PARMLIB Members

1. Copy the provided sample PARMLIB members from your installation sample library to your site's common PARMLIB dataset(s) for Compuware.

   Note: Any 17.02 or higher release of a Compuware mainframe product requires the Compuware Mainframe Services Controller (CMSC) of ECC 16.05 **with all current maintenance applied**.

Step 2. Update the CMSC with PARMLIB Information

1. Make your product's PARMLIB members available in the **//CWPARM concatenation of the Compuware Mainframe Services Controller (CMSC**).

2. After making changes to your PARMLIB members, use the z/OS MODIFY command to update the CMSC with the changes you made to the PARMLIB member.

   Note: The MODIFY command requires proper OPERATOR authority.

   The verb MODIFY can be abbreviated, **F**.

   Refreshing all PARMLIB Members:

   ```
   F cmscname,PARMLIB REFRESH
   ```

   Refreshing a Single Parameter Member:

   ```
   F cmscname,PARMLIB REFRESH member_name
   ```

   For example, you updated the parameters in member FACM00:

   ```
   F cmscname,PARMLIB REFRESH FACM00
   ```

Initialize the zIIP Enablement Service (Optional)

Overview

IBM has provided a mechanism for Independent Software Vendors (ISV), such as Compuware, to execute portions of ISV product code on zIIP processors at your site. Work executed on a zIIP processor is not charged for usage as with general use processors. As a result, when portions of Compuware product code are offloaded to execute on a zIIP processor, it can result in reduced costs for the customer.

The CSS zIIP Enablement Service provides a common method to enable selected portions of Compuware product code to be eligible to run on a zIIP processor. If available on your system, performance of Compuware software products can be enhanced by executing selected portions of the product on a zIIP processor. Execution on a zIIP processor can result in less billable CPU time used and significantly fewer I/O operations for normal product operation.

The CSS Cobol, PL/I, and Assembler language processors are particularly well-suited for zIIP processing. Executing the language processors with zIIP processing enabled can reduce CPU time used by 35 to 65% depending on the size and nature of the compiler listing being processed.

Enabling the Service

Before the service can be used, it must be started to enable its API for use by other Compuware products. The service will be started when the CMSC is started, if ZIIP_ENABLEMENT=YES. It remains active as long as the system remains running. The service can optionally be stopped, if desired.

Upon successful initialization, the following message will appear in the job log:

```
CXZPINIT010I Initialization successful for ID CPWR
```

Please refer to the *Enterprise Common Components Messages and Codes* manual for information regarding return codes.

Disabling the Service

This service can be temporarily disabled on a job-by-job basis, or completely disabled for the entire system.

Job Disablement

To disable the service for a specific job, use the CXZPIGNR DD statement.

This statement will cause the job to execute without using the service. Note that the portions of the product that were designated for zIIP processing will still be executed normally, but not on a zIIP processor.

```
//CXZPIGNR DD DUMMY
```

System-wide Disablement

If you wish to disable the service for all jobs, use the initialization utility with an EXEC PARM field to specify the keyword “DELETE”, as shown in the example below:

After running this JCL, the service will no longer be active in the system. The initialization utility can be re-executed to re-enable the service, if desired.

Chapter 6.
LMS Customization

This chapter provides the basic procedures for setting up and customizing Compuware's License Management System (LMS) for use with the Compuware products licensed by your organization. It includes instructions for:

- Transferring a License Certificate to your host system.
- Setting up and using the License Administration Utility (LAU) to create a License File and define system nodes.
- Importing a Certificate into a License File.
- Initializing the LMS runtime environment.

This chapter provides only the most basic elements needed to establish a Compuware LMS environment.

Preliminary Considerations

Which tasks you need to perform will depend on whether LMS has already been enabled at your site and whether you want to manage your Compuware license certificate from a central location or manage them individually on each of your LPARs.

The following is a list of items you might want to consider:

- Managing your licenses from a centralized location
- Running more that one license management system
- Managing more than one Compuware customer number

Migration Considerations

The following items should be considered when upgrading from an earlier release of LMS to LMS release 16.05 or later:

- Starting with release 16.05, LMS runs under the new Compuware Mainframe Services Controller (CMSC) address space. The initialization job(s) no longer run independently. By default LMS is started in the CMSC in non-CLF mode. If you wish License Management System (LMS) to run in Centralized License Facility (CLF) Mode see the "CLF" parameter found under "Step 1: Configure the CMSC Start-Up Parameters" on page 5-2.
- LMS will now get start-up parameters from the new Compuware Common Parmlib facility that's part of CMSC. Refer to Chapter 5, "Compuware Mainframe Services Controller (CMSC)" for more information.
- You must delete the earlier version's subsystem before starting the CMSC with the current LMS by specifying FUNCTION(DELETE) in the LMS Client parmlib member—the older subsystem will be deleted.

  Note: Any products that were checked out or running when the earlier version's subsystem was deleted should be shutdown and restarted. Also, any products that are started between the time the earlier version's subsystem is deleted and LMS is started in the CMSC will fail with a license error.

- The LMS 4.0 checkpoint dataset is fully compatible with subsequent LMS releases. LMS will automatically create the checkpoint dataset using the name you supplied in the LMCL00 and LMSV00 PARMLIB members if the dataset specified does not exist.
- Ensure that the LMS EXIT_PROC passes the correct parameter format:

  ```
  EXEC PGM=LMSENMGR,PARM=&LMSSYS
  ```

  Be especially careful to insure that the //STEPLIB DD in the LMS EXIT PROC points to the new release 16.05 (or later) datasets. Often this step is overlooked and results with the LMS EXIT PROC for the older release unable to update the checkpoint dataset created by LMS 16.05.

Which Steps Do I Need to Perform?

The choice of which steps to perform depends on whether LMS has already been installed and initialized as part a previous Compuware product installation. For this reason, the procedures in this chapter are tailored for two possible scenarios:

- You need to install and set up LMS for the first time, then enable it to work with the Compuware product being installed.
- You are an established LMS user who only needs to enable it to work with the Compuware product being installed.

As shown in Table 6-1, a first-time user must perform every step. Most established LMS users can skip Steps 2, 3, 4, and 5.

Table 6-1. LMS Steps Required

First-Time User	Step	Established User
x	"Step 1. Transfer License Certificate to Host"	x
x	"Step 2. Set Up License Administration Utility (LAU)"	
x	"Step 3. Create License File" on page 6-6	
x	"Step 4. Verify License File" on page 6-8	
x	"Step 5. Define Nodes" on page 6-9	
x	"Step 6. Import License Certificate" on page 6-11	[a]x
x	"Step 7. LMS Parameter Customization" on page 6-13	x
x	"Step 8. Continue ECC Customization" on page 6-14	x

a.This step is required only if new license certificates have been received.

Enabling LMS

Follow the instructions below that apply to your situation.

Step 1. Transfer License Certificate to Host

LMS uses License Certificate files to configure access to Compuware products licensed by your organization. A License Certificate is a text file typically sent to your site via e-mail by Compuware's Worldwide License Management team. To be used by LMS, a License Certificate must be accessible to MVS.

1. Locate the License Certificate for the Compuware products being installed. A License Certificate file can contain licensing for more than one Compuware product.
2. Allocate a target dataset on the host system for the License Certificate. Use the DCB parameters RECFM=FB, LRECL=80, and whatever BLKSIZE you prefer.
3. Transfer the License Certificate to the host using File Transfer Protocol (FTP), IND$FILE, cut and paste, or any other method desired.

   Note: If you open the License Certificate in an ISPF editor or transfer it using cut and paste, make sure NUMBERS is set to OFF.

4. If you have more than one License Certificate file, repeat the process as required.

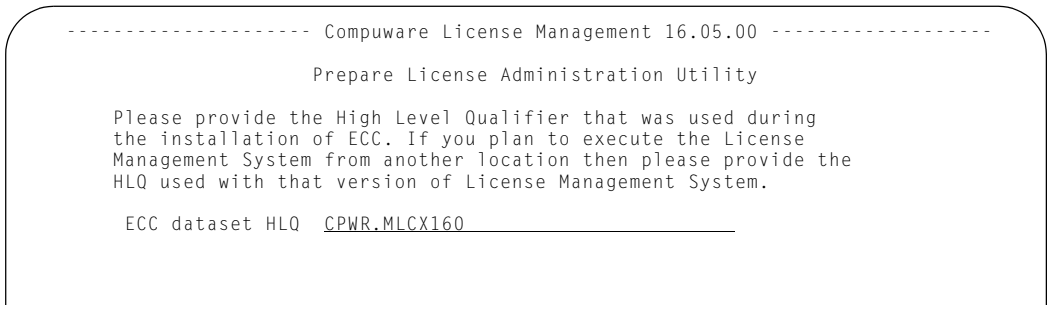
Step 2. Set Up License Administration Utility (LAU)

This step creates the CLIST for starting the LAU, prepares the LAU to run, and makes it accessible to the License Administrators at your site.

1. Enter the following command to execute the LMSPREP EXEC from the SLCXCNTL library.

   ```
   TSO EX ' your SMP/E dataset prefix.SLCXCNTL(LMSPREP)'
   ```

Figure 6-1. Prepare License Administration Utility Screen



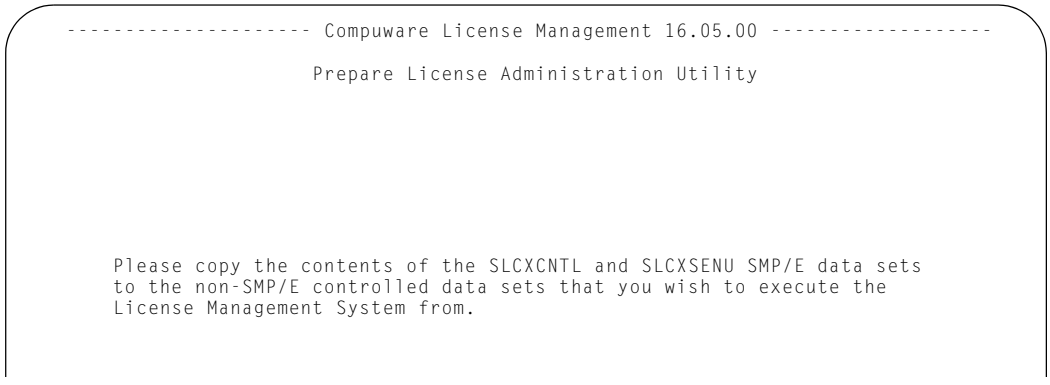
The following messages will notify you that you have successfully tailored the SLCXCNTL screen. Message LMP100I will be shown for each successfully tailored dataset. Error messages will be displayed for any unsuccessful execution.

An example of the list of messages is shown below. You may have more messages in your particular environment-based upon features chosen:

```
LMP100I SLCXCNTL(CWLMA) SUCCESSFULLY TAILORED
LMP100I SLCXCNTL(CWLMQURY) SUCCESSFULLY TAILORED
LMP100I SLCXCNTL(CWLMDMSG) SUCCESSFULLY TAILORED
LMP100I SLCXCNTL(CWLMCHKP) SUCCESSFULLY TAILORED
LMP100I SLCXCNTL(LMUSER) SUCCESSFULLY TAILORED
LMP100I SLCXCNTL(ROIMAINT) SUCCESSFULLY TAILORED
LMP100I SLCXSENU(LMASTPLB) SUCCESSFULLY TAILORED
```

2. The second screen will be displayed only when the dataset prefix field displayed in the first screen was changed to a value that did not match the dataset prefix of the SLCXCNTL library that contained LMSPREP. This is to accommodate customers who do not want to execute the LAU directly from the SMP/E target libraries for the License Management System.

Figure 6-2. Prepare License Administration Utility - second screen



3. Perform either of the following:
   - Add the SLCXCNTL library to your SYSPROC concatenation, or
   - Copy member CWLMA from the SLCXCNTL library into a dataset already concatenated to your SYSPROC DD.
4. Add the following line to an ISPF selection panel of your choice:

   ```
   L,'CMD(%CWLMA) NOCHECK'
   ```

   Note: Adding CWLMA to ISPF is optional, and you may execute CWLMA as a CLIST.

5. If your site has elected to secure the ISPF Command Table via an external security manager (i.e., RACF, TOPSECRET, or ACF2), then you must take appropriate steps to identify the command LMAMAIN to your external security manager.
6. If you are an established user, go to "Step 4. Verify License File" on page 6-8.

Step 3. Create License File

If you have previously installed Compuware's License Management System software, go to "Step 4. Verify License File" on page 6-8.

In this step, you will use the LAU to create a License File.

1. Start the LAU by selecting the ISPF menu item that was added as part of "Step 2. Set Up License Administration Utility (LAU)" on page 6-4. The License File Selection screen (Figure 6-3) will be displayed.

Figure 6-3. License File Selection Screen

```
---------------------- Compuware License Management 16.05.00 -- Row 1 to 2 of 2
Command ===>                                                  SCROLL ===> PAGE
                            License File Selection

Current Selection:
Enter New DSN  . .
                  (fully qualified without quotes)
   Delete/Define . N (Y|N)

 OR select below:-

Action     DSN                                          Added by

****************************** Bottom of data ********************************
```

2. Type the name you want to use for your License File (fully qualified without quotes) in the Enter New DSN field.
3. Type **Y** in the Delete/Define field.
4. Press Enter. The Delete/Define and Initialize License File screen (Figure 6-4) will be displayed, and the file name you specified above will be shown in the New License File field.

Figure 6-4. Delete/Define and Initialize License File Screen

```
---------------------- Compuware License Management 16.05.00 -- Row 1 to 2 of 2
Command ===>                                                  SCROLL ===> PAGE
                    Delete/Define and Initialize License File

New License File . : TSOID01.License.File.New

 Unit . . . . . . . . _________   (required for JES3 only)
Volume Serial  . . . _______     (blank for system determined volume)

Edit JCL . . . . . . Y  (Y-Yes,N-No)

     Press END Key to skip this process

Jobcard:
//Job Card information line 1
//Job Card information line 2
//Job Card information line 3
//Job Card information line 4

Select     Node      Description

****************************** Bottom of data ********************************
```

5. Type **Y** in the Edit JCL field.
6. Type the information for your jobcard in the four lines of the Jobcard field.

Note: If you are not using all four lines of the job card, comment out the unused lines.

7. Press Enter. The Edit Delete/Define JCL screen (Figure 6-5) will be displayed.

Figure 6-5. Edit Delete/Define JCL Screen

```
 File Edit Confirm Menu Utilities Compilers Test Help
------------------------------------------------------------------------------
EDIT       SYS98274.T095731.RA000.TSOID01.R0111344            Columns0000100080
Command ===>                                                  Scroll ===>CSR
****************************************** Top of Data ***************************
000001 //Job Card information line 1
000002 //Job Card information line 2
000003 //Job Card information line 3
000004 //Job Card information line 4
000005 //*
000006 //*********************************************************************
000007 //* DELETE/DEFINE COMPUWARE LICENSE FILE
000008 //*
000009 //* JCL GENERATED BY TSOID01 ON 2001-05-17 AT 15:22
000010 //*********************************************************************
000011 //*
000012 //LFDELDEF EXEC PGM=IDCAMS
000013 //*
000014 //SYSPRINT DD SYSOUT=*
000015 //SYSIN DD *
000016 DELETE (TSOID01.License.File.New)
000017 SET MAXCC = 0
000018 DEF CL(NAME(TSOID01.License.File.New)
000019       IXD
```

8. Make any necessary edits, then enter the END command. A Confirm Submission window will be displayed over the Delete/Define and Initialize License File screen.
9. Type **Y** in the Confirm Submission window and press Enter. The JCL to create your License File will be submitted.
10. Enter the END command. The License File Selection screen (Figure 6-3) will be displayed.
11. After your job has completed with a good return code, type **S** in the Action field next to your new License File and press Enter. Your License File name will appear in the Current Selection field.
12. Enter the END command.
    - If you are a first-time user, the main Compuware License Management screen (Figure 6-8 on page 6-9) will appear.
    - If you are an established user, the Parameter Option screen (Figure 6-9 on page 6-9) will appear. To access the main Compuware License Management screen (Figure 6-8 on page 6-9), enter the END command.

Step 4. Verify License File

In this step, you will use the LAU to verify which License File you will be using.

1. Start the LAU by selecting the ISPF menu item that was added as part of "Step 2. Set Up License Administration Utility (LAU)" on page 6-4. Because the LAU has been run before and a License File already exists, the main Compuware License Management screen shown in Figure 6-8 on page 6-9 will be displayed. If the dataset name of the License file that you want to use is displayed next to the Browse option, go to "Step 5. Define Nodes" on page 6-9.
2. Type 0 in the Option field and press Enter. The Parameter Option screen (Figure 6-6) will be displayed.

Figure 6-6. Parameter Option Screen

```
---------------------- Compuware License Management 16.05.00 -------------------
Option ===>

1)   Select      License File - TSOID01.LICENSE.FILE
2)   Node        Specify System Nodes




X) Exit          Return to previous panel




     (C) Copyright 2016, Compuware Corp. All Rights Reserved.
```

3. Type 1 in the Option field and press Enter. The License File Selection screen (Figure 5-7) will be displayed.

Figure 6-7. License File Selection Screen

```
---------------------- Compuware License Management 16.05.00 -- Row 1 to 2 of 2
Command ===>                                                  SCROLL ===> PAGE
                           License File Selection

Current Selection:
Enter New DSN  . .
                (fully qualified without quotes)
   Delete/Define . N (Y|N)

OR select below:-

Action     DSN                                          Added by
  _        LICENSE.FILE1                                USERID1  2000-08-07
  _        LICENSE.FILE2                                USERID1  2000-08-07
****************************** Bottom of data ********************************
```

4. Type **S** in the Action field next to the file that you want to use and press **Enter**. Your selected file should appear in the current selection field.

Step 5. Define Nodes

If your installation does not consist of multiple nodes in a JES network, or if you do not require the ability to route License Management work (Import, Export, or Reporting) from one JES node to another in your network, go to “Step 6. Import License Certificate” on page 6-11.

In this step, you will use the LAU to define (to LMS) the network nodes on which License Management maintenance or reporting tasks could be dispatched. If you are unsure of nodes defined on your system, system PARMLIB member JES2PARM contains NODE statements defined to JES.

Figure 6-8. Main Compuware License Management Screen

```
---------------------- Compuware License Management 16.05.00 --------------------
Option ===>

 0)    Parameters Specify System Parameters
 1)    Browse     License File - TSOID01.License.File
 2)    Update     License File
 3)    IMPORT     Import License Certificate
 4)    EXPORT     Export License Certificate
 5)    Reports    Run Reports

 6)    Disaster   Enable Disaster Site
 7)    Emergency  Emergency Password

 X)    Exit License Management



        Copyright (c) 2016 Compuware Corporation. All Rights Reserved.
Unpublished rights reserved under the Copyright Laws of the United States.
          Enter HELP for Copyright/Trade Secret Notice information.
```

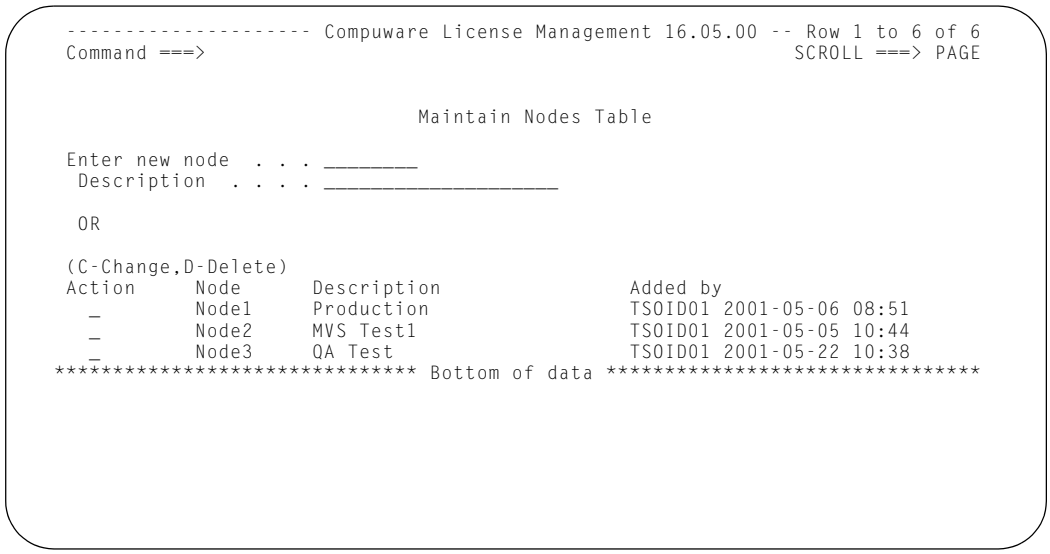
1. Type **0** in the Option field and press Enter. The Parameter Option screen (Figure 6-9) will be displayed.

Figure 6-9. Parameter Option Screen

```
---------------------- Compuware License Management 16.05.00 --------------------
Option ===>

 1)   Select      License File - TSOID01.LICENSE.FILE
 2)   Node        Specify System Nodes




 X) Exit          Return to previous panel







      (C) Copyright 2016, Compuware Corp. All Rights Reserved.
```

2. Type **2** in the Option field and press Enter. The Maintain Nodes Table screen (Figure 6-10) will be displayed.

Figure 6-10. Maintain Nodes Table Screen



3. In the Enter new node field on the Maintain Nodes Table (Figure 6-10), specify the name of the network node you want to add to the display list. This name identifies your installation's local JES in a network of systems or system complexes being used for network job entry (NJE) tasks. The node name can be up to eight alphanumeric characters.
4. Describe the new node briefly in the Description field. This description can be up to 20 alphanumeric characters.
5. Press Enter. The screen will be redisplayed with the new node in the selection list at the bottom of the screen. Repeat tasks step 3 through step 5 until you have defined all of the required nodes.
6. Enter the END command. The Parameter Option screen (Figure 6-9) will be displayed.
7. Enter the END command again. The main Compuware License Management screen (Figure 6-8) will be displayed.

Step 6. Import License Certificate

In this step the License Certificate for the Compuware product being installed will be imported into your site's License File.

1. On the main Compuware License Management screen (Figure 6-8 on page 6-9), type **3** in the Option field and press Enter. The first of two Import License Certificate screens (Figure 6-11) will be displayed.

Figure 6-11. First Import License Certificate Screen

```
---------------------- Compuware License Management 16.05.00 -------------------
Command ===>

                            IMPORT License Certificate

Process DSN  . : TSOID01.License.File

IMPORT From  . . LICENSE.CERTIF____________________
                                          (fully qualified without quotes)
Preview Only . . N (Y|N)

Reminder     . . N (Y|N)
```

2. Type the name of the dataset used for your License Certificate file (fully qualified without quotes) in the IMPORT From field.
3. Type **N** in the Preview Only field.

   Note: If you prefer to generate and review a report detailing the implications of the License Certificate importation, enter Y in the Preview Only field. You then need to repeat this step with Preview Only set to N.

4. If you type **Y** in the Reminder field you will see the message. If you type **N**, the reminder message will not appear.
5. Press ***Enter***. The second Import License Certificate screen (Figure 6-12) will be displayed.

Figure 6-12. Second Import License Certificate Screen

```
---------------------- Compuware License Management 16.05.00 -- Row 1 to 5 of 6
Command ===>                                                  SCROLL ===> PAGE

                            IMPORT License Certificate

Process DSN  . : TSOID01.License.File
IMPORT From  . . LICENSE.CERTIF
Preview Only . . N
Reminder     . . N

Edit JCL . . . . Y (Y-Yes,N-No)

Jobcard:
//Job Card information line 1
//Job Card information line 2
//Job Card information line 3
//Job Card information line 4

Select     Node      Description
  _        Node1     Production
  _        Node2     MVS Test1
  _        Node3     QA Test
****************************** Bottom of data ***********************
```

6. Type **Y** in the Edit JCL field.

7. Type the information for your job card in the four lines of the Jobcard field.

   Note: If you are not using all four lines of the job card, comment out the unused lines.

8. Type **S** next to the nodes on which you want the Import job to run. If no node is selected, the job is submitted without any specific routing.

   Note: If you are running on only one node, do not make a selection here.

9. Press Enter. The Edit JCL screen (Figure 6-13) will be displayed.

Figure 6-13. Edit JCL Screen

```
 File Edit Confirm Menu Utilities Compilers Test Help
--------------------------------------------------------------------------------
EDIT SYS98274.T095731.RA000.TSOID01.R0111344              Columns0000100080
Command ===>                                               Scroll ===>CSR
********************************** Top of Data ******************************
000001 //Job Card information line 1
000002 //Job Card information line 2
000003 //Job Card information line 3
000004 //Job Card information line 4
000005//*
000006//*********************************************************************
000007//* IMPORT COMPUWARE LICENSE CERTIFICATE
000008//*
000009//* JCL GENERATED BY TSOID01 ON 2001-07-08 AT 15:32:17
000010//*********************************************************************
000011//*
000012//IMPORT EXEC PGM=LMALFIM,
000013// PARM='UPDATE'
000014//STEPLIB DD DSN=LM.DEVL.LOAD,
000015// DISP=SHR
000016//*
000017//CWLF0000 DD DSN=TSOID01.License.File,
000018// DISP=SHR
000019//CWLFIMP DD DSN=LICENSE.CERTIF,
```

10. Make any necessary edits, then enter the END command. A Confirm Submission window will be displayed over the second Import License Certificate screen.
11. Type **Y** in the Confirm Submission window and press Enter. The JCL to create your License File will be submitted.
12. Enter the END command. The main Compuware License Management screen (Figure 6-8 on page 6-9) will be displayed.
13. Type **X** in the Option field and press Enter to exit the LAU.
14. Proceed as follows:
    - If you are a first-time user, continue with “Step 7. LMS Parameter Customization”.
    - If you are an established user, go to Chapter 7, “HCI Customization”

Step 7. LMS Parameter Customization

1. In order to function correctly, the LMS load modules must reside in an MVS APF authorized library. Consult your site's MVS system programmer for details.

   Parameters are read from the common PARMLIB dataset. Edit the sample provided in SLCXCNTL(LMCL00) (see "Sample LMS Parameters Dataset").

   - The operand and its associated value can start in any column and can be continued on more than one statement.
   - Columns 1 through 71 are available for use.
   - Column 72 is reserved.
   - Columns 73 through 80 can be used for sequence numbers. They are ignored by LMS.
   - To continue an operand on a second statement, complete the first statement through column 71, and continue the operand on the next statement starting in column 1.
   - Only the value specified for SUBSYSTEM_ID is case-sensitive. All other operands and their values are folded to upper case by LMS.
   - Only one occurrence of each of the operands is allowed. If duplicates exist, the last one encountered will be used.
   - An asterisk '*' in column 1 causes the entire statement to be a comment.
   - Line level comments are supported using the '/*' to start a comment and '*/' to end the comment. Embedded comments are supported.

   Note: The following parameters are required: FUNCTION(UPDATE), SITE(), SUBSYSTEM_ID(), CHKPT_DSNAME(), EXIT PROC().

   Additional Note: ROI_CAPTURE() is recommend by Compuware and turned on by default. SMF_ID() is also required for ROI and must contain an SMF Record ID.

Sample LMS Parameters Dataset

You will find a sample LMS parameter dataset in the sample dataset shipped to you and installed with ECC. The member name is LMCL00 and should be used as a skeleton for parameters for you particular installation. Since this member can change with PTF maintenance, it is not listed here.

In releases of LMS prior to 16.05, the LMINPARM member found in the *hlq*.SLMSCNTL library was customized. For current release of ECC, the *hlq*.SLCXCNTL library contains an additional module, LMCL00, which needs to be customized. This module reads from the common parmlib dataset. If you are an established user, compare the existing LMINPARM member with the LMCL00. You should only see one new parm: LICENSE_DSNAME. **You may be able to simply copy the previous parm into the LMCL00 member if they are the same. If you copied your previous release's LMINPARM member to LMCL00, add the LICENSE_DSNAME parm, in which you should include the name of your license file.** However, please be aware of any additional parms that may be added due to maintenance for ECC release 17.02.

2. To ensure access to Compuware products is automatically enabled, establish a procedure in your SYS1.PROCLIB to launch your CMSC as a started task during IPL and IML processing (see "Running the CMSC" on page 5-1). Consult your site's MVS system programmer for details. Edit the sample proc provided in SLCXCNTL(CMSC). Change the dataset names to match those used at your site, including the PARMLIB dataset you saved your site specific LMS parameter dataset.

3. For non-CLF, or local installations, specify the license file dataset name in the LMCL00 module, under the LICENSE_DSNAME parameter.
4. If you changed your LMS PARMLIB member name from the default LMCL00 (for example: LMCL0005) update your CMSC PARMLIB member to contain the suffix of the new LMS PARMLIB member, `LMCL=0005`. Refer to "Step 1: Configure the CMSC Start-Up Parameters" on page 5-2.

Step 8. Continue ECC Customization

If you have any Compuware products that use Host Communication Interface (HCI), you must continue with the procedures in **Chapter 7, "HCI Customization".**

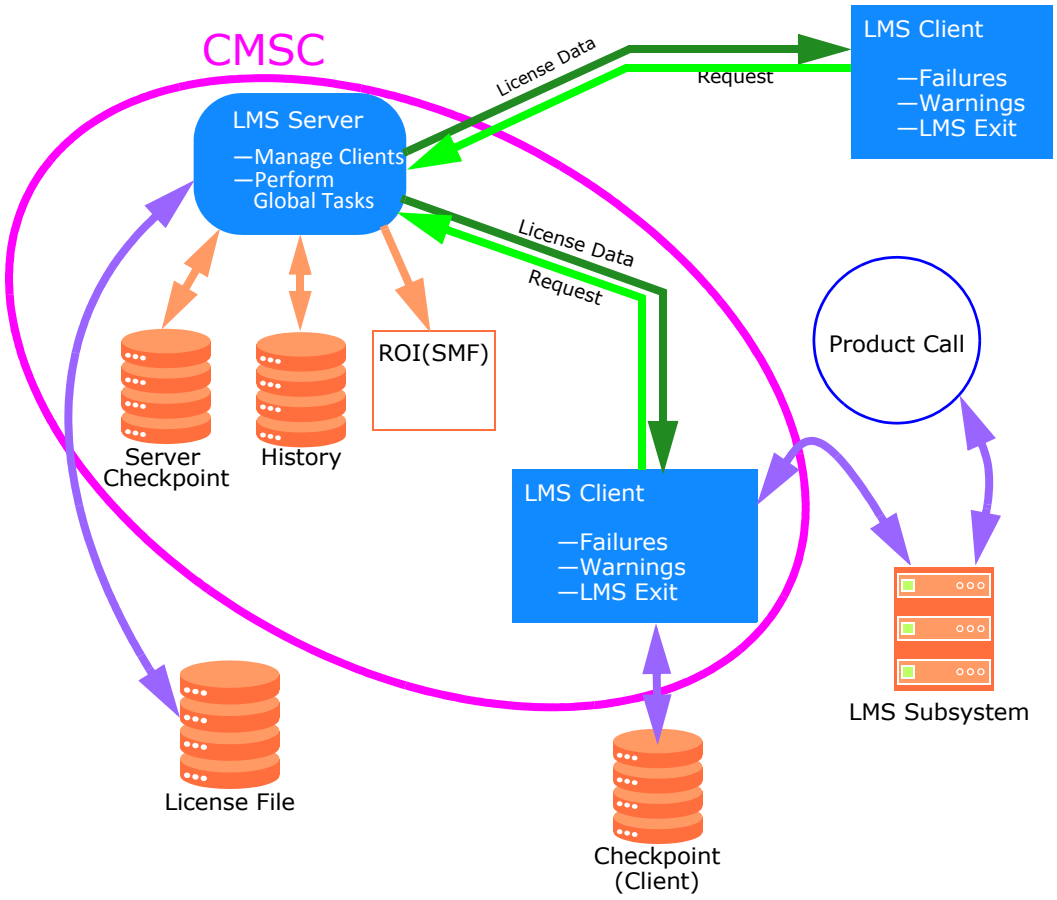
If you are installing Abend-AID or Abend-AID for CICS, refer to the respective installation and customization guides for configuration instructions concerning the Base Services and HCI components in the ECC libraries.

Resolving Problems

If any error messages were returned while performing steps in this chapter, refer to the *Enterprise Common Components Messages and Codes* guide.

If you encountered any other difficulties related to License Management System, please consult Compuware Customer Support.

Centralized License Facility (CLF) Flowchart



Chapter 7.
HCI Customization

Skip this chapter if you do not have Compuware products that require HCI or if HCI has already been installed and configured on this LPAR for use with your Compuware product. Proceed directly to Chapter 8, "CSS Customization" to customize the Compuware System Services (CSS).

This chapter contains important information that can help you install the mainframe components needed to support the host communication from the mainframe. Please review this chapter before beginning the actual installation process.

IMPORTANT: For the most up-to-date system requirements and installation instructions, please consult the current *Release Notes* by visiting the Compuware Go customer support Web site at https://go.compuware.com.

Introduction

In order to facilitate communication with a z/OS LPAR, the communications software Host Communications Interface (HCI), must be installed and configured on that LPAR. This chapter discusses the *minimum* number of configuration parameters necessary to facilitate communication with the z/OS LPAR. Depending on the Compuware mainframe applications and functionality you plan to use, additional CSS TP parameters may need to be specified. Detailed explanations of each CSS TP configuration parameters are provided in the Installation manuals of the products requiring them.

The ECC media provides an automated Mainframe upload and SMP/E installation facility contained within the Media browser. The automated installation facility for ECC, uploads, restores, and performs an SMP/E installation of the ECC software. During the automated installation process, a dataset, *hlq*.CYnnnnnn.SMPE.DATA is created containing the dataset names and other variables related to your installed ECC software.

The Compuware Shared Services mainframe Transaction Program (CSS TP) serves as the primary contact point between the mainframe and certain Compuware products. The services provided include job submission, output viewing, dataset retrieval and update, launching of Xpediter-based debugging sessions, initiating sub-tasks to support File-AID/Eclipse or File-AID Data Privacy sessions, and many other services.

For Strobe sites that use HCI, use the same port number to configure HCI for Compuware Workbench. The HCI you configure for Compuware Workbench will then run instead of the HCI you have set up for Strobe and it will be transparent to Strobe users.

For DevEnterprise sites that use HCI, you may optionally merge the HCI you configure for Compuware Workbench with the HCI you have configured for DevEnterprise.

Preliminary Considerations

Compuware's Host Communication Interface (HCI) and Compuware Shared Services (CSS) are required in order to communicate with a mainframe host.

Important: To communicate with more than one LPAR, repeat the procedures in this chapter to configure HCI for each LPAR.

You need to know the dataset names of the following SMP/E controlled libraries:

- ECC load library (SLCXLOAD)[1]
- ECC load library (SLCXAUTH)[1]

Note: When using the Compuware Installer to generate the JCL to SMPE install ECC, tables are created containing these dataset names.

Note: The HCI started task requires that an OMVS segment be defined for the user ID associated with the HCI started task in order to perform various TCP/IP services.

Connection to z/OS Host

Before you install the Host Communication Interface (HCI) on the z/OS host, you must be able to run TCP/IP traffic from the workstation to the host.

To avoid any unpredictable connection issues, make sure all current IBM maintenance is applied to TCP/IP.

Collect TCP/IP Information

- TCP/IP region name
- TCP/IP host name or IP address
- TCP/IP port number

Dispatching Considerations

It can be difficult to determine the "pecking order" of task execution in an MVS system without knowing the functions of the tasks. The HCI is much like a multiple-user region because it services many other regions much like VTAM. But the HCI itself uses resources like VTAM and VSAM. Its position in the pecking order (i.e. dispatching priority) should be set so that it is below VTAM, VSAM, and other communications subsystems (for example, TCP/IP) but above the regions that it services.

Migration Utility for HCI 3.0 XML Parameters

If you are upgrading from HCI Release 3.0 to Release 16.05, Compuware provides a utility that converts HCI Release 3.0 XML parameters and their associated CSS TP configuration parameters into a PARMLIB member for use with ECC Release 16.05.

Modify the sample JCL (HCIJMGRT), located in SLCXCNTL, to execute the migration utility. The utility writes its output into the specified PARMLIB member, for example HCI00, replacing any data in the member. The output includes the parameters and their values.

1. Specify your job card.
2. Update the DD definitions with your site's library names, fully qualified without quotes:
   a. //HCIXML - Specify the HCI Release 3.0 member that contains the HCI XML parameters to be converted.
   b. //TPCONFIG - Specify the CSS Release 9.0 library that contains the TP configuration members.
   c. //HCIPARM - Specify the pre-allocated PARMLIB dataset and the member to write the converted data to.
   d. //SYSTSIN - Specify the library member HCIRMGRT in the EXEC statement that contains the modified JCL.

1. In previous releases of ECC, SLCXLOAD and SLCXAUTH were associated only with CSS. However, for ECC 16.05 or higher, those libraries are used by all of ECC product facilities, including HCI, CSS, and LMS.

3. Submit the job.
4. Review the output PARMLIB member.
5. Make any changes according to your site's requirements for the current release of ECC. Refer to "Step 2. HCI Parameter Customization" on page 7-3 for more details.

   If you intend to run multiple HCI instances concurrently, make sure to avoid duplicate SYSID and port values.

Note: If you are upgrading from a HCI Release prior to 3.0 you have to manually update the PARMLIB members with your site's values.

Step 1: HCIJOURN

Run the HCIJOURN job, which allocates the HCI journal datasets and the CSS TP journal datasets. The CSS TP journals are only necessary if you are going to deploy the SSAS address space to enable Compuware DDIO file support of the Topaz Workbench's Code Coverage/Eclipse plug-in (see the *Topaz Workbench Installation Guide*).

Step 2. HCI Parameter Customization

Specify the Minimum HCI Parameters

Located in SLCXCNTL, the HCI00 Parameter File contains customization parameters for HCI, CSS TP and (optionally) DevEnterprise. A copy of this member must reside in the //CWPARM concatenation of the CMSC (see Chapter 5, "Compuware Mainframe Services Controller (CMSC)" for more information.

In the HCI00 Parameter File, specify the following parms:

SYSID

Specifies the four-character subsystem name of HCI. You must ensure this name is unique among all subsystems running on the LPAR, and *not* defined in SYS1.PARMLIB. The HCI dynamically creates this subsystem for you. Each HCI that you desire to run on a single LPAR must have its own unique SYSID value.

DEFAULT_USER

Specifies the name of a valid RACF (ACF/2 or TOPSECRET) user ID to be used by the HCI when TP jobs are submitted or started and when there is no other user ID available. Ensure that the default user ID has either ACCESS(READ) or ACCESS(EXECUTE) to the //STEPLIB DD used in the TP JCL and that it has appropriate access to any other datasets that it processes under the default user ID. Once the TP has issued a security call, the default user ID is no longer used for access as the TP starts running under the user ID passed in the security call. Must be a valid USERID.

OPCMD

How operator commands are entered to the HCI.

```
OPCMD=[WTOR|MODIFY]
```

WTOR — Reply to Operator (r nn,) is to be used

MODIFY — z/OS MODIFY (/F) is to be used

JOURNALn

Specifying journal datasets allows for logging of HCI activity. The HCI uses the Journaling Facility to write out diagnostic information that may then be provided to

Compuware Developers. **Note:**Journal datasets cannot be shared across multiple HCI instances. Each HCI instances requires its own set of journal datasets.

CPWR_PORT

Specifies a port number in which the HCI listens for incoming connection requests. Multiple entries may be specified, which correspond to a given PORT_CONFIG entry located under the CSS TP PORT CUSTOMIZATION section of the HCI parameter file.

Valid port numbers range from 1 through 65535. Check with your network administrator for available port numbers.

Specify the Minimum CSS TP Parameters

For products requiring CSS TP parameters, details are located in those products' installation guides (for example, the "TP Configuration Member" appendix in the *Topaz Workbench Installation Guide*). The CSS TP parameters section begins at the PORT_CONFIG location in the HCI00 Parameter File. Order must be retained for composite keywords.

Specify the following:

PORT_CONFIG

Denotes the start of a CSS TP port configuration. Everything below this keyword will apply to the value specified for PORT_CONFIG.

STASK

The CSS TP Started Task procedure name. The STASK is required for Endevor, File-AID Data Privacy and/or File-AID/Eclipse plug-in support.

SSAS_PROC

The Shared Services Address Space procedure name. The SSAS_PROC statement identifies the JCL procedure name used by the Compuware Shared Services Address Space (SSAS).

SSAS_NAME

The Shared Services Address Space. The SSAS_NAME identifies the address space name to be used by the Compuware Shared Services Address Space (SSAS). This address space is intended to run unauthorized functions on behalf of Topaz Workbench.

LOCAL_PORT=nnnnn

LOCAL_HOST=&LPARNAME..COMPUWARE.COM

LOCAL_ZIIP=Y

The Local statements identifies the HCI using this PORT_CONFIG configuration, and whether select TP activity should be executed on an available zIIP processor. If required, multiple LOCAL_* statements may be specified.

HCI_SYSNAME=&LPARNAME.-TP-PROD

HCI_LPAR=&LPARNAME

HCI_PORT=nnnnn

HCI_HOST=&LPARNAME..COMPUWARE.COM

Service HCI definition. This allows for support of Host Explorer connections to the specified LPAR.

for Strobe Web Service Users only

The only required parameters for Strobe are LOCAL_*, STASK, and HCI_*. See "Step 3: HCIPROCS" on page 7-6 for STASK procedure requirements.

for DevEnterprise Users only

The following parameters are required to run DevEnterprise. These default parameters must be included in the HCI00 PARMLIB member Product Definitions section.

Note: These parameters must be specified before any PORT_CONFIG statements are specified.

```
DEVENT_GCS_PORT=1216              COLLECTOR GCS
DEVENT_TCP_PORT=12160             COLLECTOR TCP
TPT_NAME=XPPXFER                  -- XFER JCL START
START XPEDFT16
END
TPT_NAME=XPPJOBM                  -- JOBM JCL START
START XPEDJM16
END
TPT_NAME=XPPTCPIP                 -- TCPIP JCL START
START XPDDIO16
END
```

DEVENT_GCS_PORT=

Specify the DevEnterprise Job Manager port.

DEVENT_TCP_PORT=

Specify the DevEnterprise TCP/IP port.

TPT_NAME=

START *name*

END

Specify the *name* of a TP and command input to the HCI. Text between TPT_NAME= and END will be used as input to the HCI.

for Sysplex Support

SYSPLEX=xxxx

Four-character subsystem name, which must be different than the HCI SYSID specified. This name must be unique among all MVS images in the SYSPLEN and must not be specified in the IEFSSNxx member of SYS1.PARMLIB.

ROUTCMD=YES|NO

Yes specifies that the HCI routes z/OS START commands to the active member of the sysplex that currently has the least number of routes to it. The following example illustrates what changes would be required to enable XPPJOBM a SYSPLEX routed task.

TPT_NAME=XPPJOBM
ROUTE (CW09,CW03,CW01) XPEDJM16
END

PREPROC=procedure

This value specifies the name of the procedure in which the HCI should invoke on each MCS image in the SYSPLEX to prepare the MVS for HCI communication.

Sample PREPROC procedure that will activate SYSPLEX support in the HCI. No changes are required below aside from specifying the SLCXAUTH dataset:

```
//IEFPROC PROC HCISYSP=DUMY,HCITYPE=

//HCIPREP EXEC PGM=HCIYPREP,PARM='&HCISYSP,&HCITYPE'
```

```
//STEPLIB DD DISP=SHR,DSN=CPWR.SLCXAUTH
```

Step 3: HCIPROCS

Run the HCIPROCS job, which creates three procedures in the dataset specified by SYSUT2 (required for Topaz Workbench).

- **HCIPROC** — HCI procedure. Optionally, the HCI can be run as a batch job. Typically this is done to verify customization.
- **CXSSAS** — Shared Services Address Space (SSAS), used primarily for DDIO and Xpediter/Code Coverage.
- **CCSS0000** — CSS TP started task, used mainly by File-AID and Strobe, designed to run as unauthorized.

  IMPORTANT: If you are planning to use the File-AID Data Editor or File-AID Data Privacy, be sure to verify and uncomment the DD for the File-AID libraries in the STEPLIB concatenation (see the chapter entitled, "File-AID Product Installations and Configurations" in the *Topaz Workbench Installation Guide)*.

  Depending on your installed releases of File-AID products, you may optionally code each of the File-AID libraries in the STEPLIB concatenation of the STASK procedure. When specifying individual libraries for each File-AID product.

  For File-AID 10.1 or higher, there are no separate product libraries, there are only two required datasets CXVJLOAD and SXVJAUTH.

  Use the File-AID 10.1 or higher (File-AID 16.3 recommended) customization (CXVJLOAD) and distribution authorized load library SXVJAUTH (as shown in Figure 7-1).

  The &TPNAME is a value generated by the CSS TP internally and should not be modified.
  IMPORTANT: If using this HCI configuration to support Strobe Web Services or SQLAF, the STEPLIB datasets to be activated for Strobe are:

  `STRnnn.SSTRAUTH` **(can be in the LINKLIST)**

  `DSNxxx.SDSNLOAD` **(if not in LINKLIST).**

  for addition tracing:

  ```
  //SBSQLTRC DD DUMMY
  ```

  ```
  //SBSQLSNP DD DUMMY
  ```

Figure 7-1. SLCXCNTL member HCIPROCS Startup JCL example

```
//*HCI      JOB  0,HCI
//* ************************************************************* *//
//*                                                               *//
//* THIS JCL PERFORMS THE FOLLOWING INSTALLATION PROCESSING:      *//
//*    STEP(S)   FUNCTION                                         *//
//*    --------  -------------------------------------------      *//
//*    HCPROC    CREATE CATALOGED PROCEDURES TO SYSTEM            *//
//*              PROCEDURE LIBRARY.                               *//
//*                                                               *//
//*    NAME(S)   FUNCTION                                         *//
//*    --------  -------------------------------------------      *//
//*    HCIPROC   HOST COMMUNICATIONS INTERFACE                    *//
//*    CXSSAS    SHARED SERVICES STARTUP PROCEDURE                *//
//*    CXSS0000  SHARED SERVICES TP STARTED TASK                  *//
//*                                                               *//
//*    DATASET(S)      DESCRIPTION                                *//
//*    --------------  ---------------------------------------    *//
//*    CPWR.SLCXAUTH   ECC AUTHORIZED LIBRARY                     *//
//*    CPWR.SLCXLOAD   ECC LOAD LIBRARY                           *//
//*    SYS1.PROCLIB    PROCEDURES DESTINATION LIBRARY             *//
//*                                                               *//
//* ************************************************************* *//
//HCPROC  EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT DD  SYSOUT=*
//SYSUT2   DD  DISP=SHR,DSN=SYS1.PROCLIB
//SYSIN    DD  DATA,DLM=ZZ
./ ADD NAME=HCIPROC
//HCI      PROC
//*
//*  HOST COMMUNICATIONS INTERFACE PROCEDURE
//*
//STEP1    EXEC PGM=HCIMMAIN,DYNAMNBR=125,
//             REGION=64M
//STEPLIB  DD  DISP=SHR,DSN=CPWR.SLCXAUTH
//HCIJCL   DD  SYSOUT=(A,INTRDR)
//HCIPRINT DD  SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
//SYSPRINT DD  SYSOUT=*
//HCIERR   DD  SYSOUT=*
//ABNLIGNR DD  DUMMY
//IDIOFF   DD  DUMMY
./ ADD NAME=CXSSAS
//CXSSAS PROC
//*
//*  SSAS STARUP PROCEDURE
//*
//*  - THE HCI AUTOMATICALLY STARTS THIS PROCEDURE AS NEEDED
//*    AND SHOULD NOT BE STARTED MANUALLY.
//*  - FOR CODE COVERAGE SUPPORT UNCOMMENT THE CC LIB DD
//*
//IEFPROC  EXEC PGM=CXASINIT,
//             PARM=('&TPNAME'),    ** DO NOT CHANGE THIS PARM VALUE **
//             REGION=0M
//STEPLIB  DD  DISP=SHR,DSN=CPWR.SLCXLOAD
//*        DD  DISP=SHR,DSN=CPWR.SLXVLOAD              <== CC LIB
./ ADD NAME=CXSS0000
//CXSS0000 PROC PROG=IEFBR14
//*
//*  CSS TP STARTED TASK
//*
//*  - ALL STEPLIB LIBRARIES MUST BE APF-AUTHORIZED
//*  - FOR FILE-AID SUPPORT UNCOMMENT THE FA CUST/DIST LIB DDS
//*  - FOR STROBE   SUPPORT UNCOMMENT THE SB LIB DDS AND
//*    OPTIONALLY THE DB2 SDSNLOAD LIBRARY
//*
//IEFPROC  EXEC PGM=&PROG,
//             REGION=0M,
//             PARM=('&TPNAME')
//STEPLIB  DD  DISP=SHR,DSN=CPWR.SLCXAUTH
//*        DD  DISP=SHR,DSN=CPWR.CXVJLOAD              <== FA CUST LIB
//*        DD  DISP=SHR,DSN=CPWR.SXVJAUTH              <== FA DIST LIB
//*        DD  DISP=SHR,DSN=CPWR.SSTRAUTH              <== SB LIB
//*        DD  DISP=SHR,DSN=DSNA10.SDNSLOAD            <== DB2 FOR SB
./ ENDUP
ZZ
```

Step 4: Continue with ECC Customization

Proceed to **Chapter 8, "CSS Customization"** to customize the Compuware System Services (CSS).

Overview of Operator Commands

DISPLAY

Display status information about an HCI region.

Operands	Description
ALL	Displays all HCI status fields.
JOURNAL	Displays information about the journal.
STOR	Displays information about current storage utilization.
CB	Displays information about control block utilization. The CURRENT utilization, HIGHEST utilization, and MAXIMUM allowed as configured in the HCICNFIG file are displayed.

SHUTDOWN

Terminate the HCI.

Operands	Description
IMMED	For an immediate shutdown. The HCI will not wait for any user tasks to end prior to termination. Consequently, the HCI will end with a SA03 abend if there are any active users at shutdown time.
NORMAL	For normal shutdown
ABEND	For a forced abend situation

Additional Commands

Various commands that may be useful in controlling an HCI.

Operands	Description
CLOSE	Turn journaling off. Close and deallocate the current journal dataset.
OPEN	Allocate and open the next journal dataset.
JMASK [ON/OFF]	Turn on/off full journaling. TP activity is the default journal mask.
SWAP	Close and deallocate current journal dataset, open the next for journaling.

Chapter 8.
CSS Customization

Skip this chapter if you do not have Compuware products that require CSS. Proceed directly to Chapter 9, "Verification of ECC Component Customization" to verify the ECC customizations you have made.

The procedures in this chapter are used to customize Compuware Shared Services (CSS).

As shown in Table 8-1, some of the steps in this chapter are required, while others are optional.

Table 8-1. CSS Procedure Requirements

Step Number	Required	Optional
"Step 1. Implement the Security Exit Program (Optional)"		x
"Step 2. Install Customized Translation Tables (Optional)"		x
"Step 3. Make New CSS Load Modules Accessible to Compuware Products (Required)"	x	
"Step 4. Create the Compile Information Table (Required)"	x	
"Step 5. Activate Compuware ISPF Dialogs (Required)"	x	
"Step 6. Link CSS Utilities Tutorials to Your Main Tutorial Panel (Optional)"		x
"Step 7. Prepare the DDIO Files"	x	
"Step 8. Implement the Language Processor JCL (Required)"	x	
"Step 9. Continue with ECC Verification"		x
"Step 9. Continue with ECC Verification"	x	

Step 1. Implement the Security Exit Program (Optional)

The Security Exit program is an optional user-coded module for establishing security at each site. If you want to ensure full security access to file resources, refer to the "CSS Security Exit" chapter in the *Compuware Shared Services User/Reference Guide* for detailed information on using the program. A sample Security Exit program is located in SLCXCNTL sample library member CWASSECU.

If the Security Exit program is present, it is called for each member selected. The Security Exit program determines if the user command is permitted, and passes the appropriate return code. If the command is unacceptable for one of the selected members, it will not be executed at all.

Because CSS is used by more than one Compuware product, the Security Exit program is executed for each Compuware product that uses CSS. If you are using multiple Compuware products, you must update your Security Exit program to accommodate those products.

Notes:

1. The sample Security Exit program is for illustration purposes only and **does not** provide any useful functions. You must update it to serve the needs of your site.
2. This Security Exit program is invoked in the MVS environment and does not provide for CICS services.
3. If Abend-AID for CICS is installed at your site, or if you are installing Abend-AID for CICS, the Security Exit program must be reentrant.

To Implement the Security Exit Program:

Use the following procedure to implement the CWASSECU Security Exit program:

1. Create the Security Exit program. Use the sample Security Exit program as the basis for your exit.

   You must use the program name CWASSECU because other programs reference it and search specifically for this program name.
2. Assemble and linkedit the Security Exit program as a separate load module named CWASSECU. Linkedit CWASSECU into the same load library as CSS. The SLCXCNTL library member CXJCLSEC contains sample JCL that can be used to assemble and linkedit CWASSECU to the ECC load library.
3. If CICS Abend-AID/FX or Abend-AID for CICS is installed at your site, the Security Exit program must be reentrant.

Step 2. Install Customized Translation Tables (Optional)

Note: If you omit this step, the default translation tables will be used.

You must perform this step if you are using Japanese language support with Abend-AID.

You may optionally perform this step if you have Abend-AID. When necessary, Abend-AID products dump storage in both hexadecimal and character representation. These storage areas will appear in one of two formats:

- Horizontal dump
- Vertical dump

The character representation for either format is controlled by a default translation table. The default character set is mixed-case English. You can override either, or both, of these default translation tables with customized translation tables. Different customized translation tables can be specified for either dump format. These customized tables may also allow display of certain otherwise non-displayable fields.

If you install vertical and/or horizontal customized translation tables, CWCMTRVT must be used as the load module name for the vertical translation table, and CWCMTRHT must be used as the load module name for the horizontal translation table. A different internal name can be defined to identify a table. The internal table name must be eight characters long. The table length, excluding the table name, must be 256 bytes. Therefore, the total table length is 264 bytes. If an error is made installing a customized translation table, the Compuware default table is used.

The following SLCXCNTL sample library members contain sample translation tables:

CWCMTRVT	Vertical translation table for uppercase English.
CWCMTRHT	Horizontal translation table for uppercase English.
CWCMTRVE	Vertical translation table for mixed-case English. This is the default vertical translation table that will be used if you choose not to install a customized vertical translation table.
CWCMTRHE	Horizontal translation table for mixed-case English. This is the default horizontal translation table that will be used if you choose not to install a customized horizontal translation table.
CWCMTRVU	Vertical translation table for mixed-case English with the Euro Character X‘9F’ included.
CWCMTRHU	Horizontal translation table for mixed-case English with the Euro character X’9F’ included.

The ECC SLCXCNTL sample library member CXJCLTRT contains the sample JCL to assemble and link-edit customized translation tables.

Step 3. Make New CSS Load Modules Accessible to Compuware Products (Required)

You must perform this step.

Ensure that the new CSS load modules can be accessed by Compuware products by using one or more of the following methods. Regardless of the method chosen, the ECC load library (SLCXLOAD or a copy of it) should be concatenated **in front** of any Compuware product library(ies). The method you choose will depend on the Compuware products you have at your site.

- Compuware recommends that you place the ECC load library in the link list. This will provide access to the CSS load modules for TSO users and batch jobs that need CSS (for example, language processors, CSS utilities, and/or Abend-AID). This also minimizes any JCL changes that might be needed for a CSS upgrade. (A refresh of the link list is required to activate this change.) Refer to "APF Authorization" on page 2-4 for additional information and cautions about APF authorizations.
- If you have a supported release of Xpediter/CICS installed at your site, you are required to include the ECC load library with the Xpediter/CICS library in the CICS startup JCL or RDO definitions for each CICS region.
- You may also place the ECC load library in the STEPLIB or ISPLLIB for individual TSO logon PROCs, and in the STEPLIB or JOBLIB for batch jobs. These library specifications will override the ECC load library specified in the link list. This is useful for testing a new version of CSS prior to placing it into the link list for production use.

Notes:

1. If you have previously installed CSS and referenced the load library in batch jobs, compile PROCs, CLISTs, or logon PROCs, CSS will be loaded from the STEPLIB, ISPLLIB, or JOBLIB where it is specified rather than from the link list. Compuware recommends that you review these locations and remove or modify the CSS load library DDs as appropriate. If you have any Compuware product that uses CSS from a STEPLIB, ISPLLIB, or JOBLIB concatenation, you should include the ECC load library in front of the Compuware product in that library concatenation.

   You have the ability to dynamically allocate the ECC load library as part of the ISPLLIB concatenation. Please refer to Option 1 listed in the next step for more information.
2. For Abend-AID for CICS, specify the ECC load library names in the server JCL. Refer to the *Abend-AID Installation and Customization Guide* for more information.

Step 4. Create the Compile Information Table (Required)

You should perform this step if you plan to use the CSS Utilities or the Eclipse Compile Facility within Host Explorer (see *Topaz Workbench Installation Guide*).

The purpose of this step is to create the compile information table, which is stored in SLCXTABL. This table contains information about your compilers, compiler libraries, DB2 pre-compile libraries, CICS translator libraries and some miscellaneous items. A series of ISPF dialog panels will take you through the table creation process. When you have finished creating the table, you will have to make it available to the CSS Utilities. See "Step 5. Activate Compuware ISPF Dialogs (Required)" on page 8-15.

For the Eclipse Compile Facility, the compile information table is accessed to populate the Preprocessor Compile Libraries dialog when the Import button is selected (see Host Explorer product Help in the Topaz Workbench).

Create the Compile Information Table:

1. You must edit the CXUECIN2 member of the SLCXCNTL sample library. To avoid having your modifications overlaid by future maintenance, you may want to copy the member to a non-SMP/E controlled library that is in your SYSPROC concatenation. Follow the instructions that are in the member. You will need the name of your ECC panel library, SLCXPENU, and your ECC REXX EXEC/CLIST library, SLCXEXEC.
2. If you placed the member, CXUECIN2, in a library that is in your SYSPROC concatenation, then you may invoke the REXX EXEC as follows: `TSO %CXUECIN2`. Otherwise, you may invoke it as follows:
   `TSO EXEC '`*`your.dataset.library`*`(CXUECIN2)'`.

   The first panel displayed is the “Install Menu”. See Figure 8-1below.

Figure 8-1. Compile Information Table - Install Menu

```
Compuware CSS Utilities --------- Install Menu --------------------------------
COMMAND ===>

 Enter the name of the library, which is used to store the compile information
 that you supply from the options below.  If the library does not exist, it
 will be allocated.
 ===>

 If you have already supplied the compile information in a prior installation
 and you want to copy that information to a new library, then enter the
 'old' library name below.  Use the 'C' option to copy from the library named
 below to the library named above.
 ===>

 Enter an option on the command line.
       1.) Select compilers
       2.) Enter compiler libraries
       3.) Enter DB2 IDs and precompile libraries (optional)
       4.) Enter CICS IDs and translator libraries (optional)
       5.) Enter miscellaneous information
       A.) Process options 1 through 5
       C.) Copy a prior installation's information to a new library

          Press ENTER to process  or  enter END command to terminate
```

The first field is the name of the Compile Information table. All of the information entered will be saved in this table. If the file does not exist, then it will be allocated for you.

The second field is used during the COPY command. See “C. Copy a Prior Installation's Information to a New Library” on page 8-12.

Options:

1. Select Compilers

   This option displays the available compilers that the Compile Facility supports. Place a **Y** next to each compiler that your site uses and you want your programmers to be able to use with the Compile Facility. You must enter this option first and select at least one compiler. See Figure 8-2 on page 8-6 and Figure 8-3 on page 8-6 below.

Figure 8-2. Compile Information Table - Select Compilers (1 of 2)

```
Compuware CSS Utilities ------- Select Compilers -------------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:     +
COBOL:
  OS/VS COBOL           ===> N  (Y/N)
  VS COBOL II           ===> N  (Y/N)
  COBOL/370             ===> N  (Y/N)
  COBOL/MVS             ===> N  (Y/N)
  COBOL/390             ===> N  (Y/N)
  ENTERPRISE/COBOL      ===> N  (Y/N)
  COBOL5                ===> N  (Y/N)
  CA OPTIMIZER          ===> N  (Y/N)
    5.1 DDNames         ===> N  (Y/N)

Assembler:
  Assembler H           ===> N  (Y/N)
  High Level Assembler  ===> N  (Y/N)

PL/I:
  PL/I                  ===> N  (Y/N)
  PLI/370               ===> N  (Y/N)


          Press ENTER to process  or  enter END command to terminate
```

Figure 8-3. Compile Information Table - Select Compilers (2 of 2)

```
Compuware CSS Utilities ------- Select Compilers -------------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:   -
  CA OPTIMIZER          ===> N  (Y/N)
    5.1 DDNames         ===> N  (Y/N)

Assembler:
  Assembler H           ===> N  (Y/N)
  High Level Assembler  ===> N  (Y/N)

PL/I:
  PL/I                  ===> N  (Y/N)
  PLI/370               ===> N  (Y/N)
  VA/PLI                ===> N  (Y/N)
  ENTERPRISE/PLI        ===> N  (Y/N)

C Language:
  C/MVS                 ===> N  (Y/N)
  C/390                 ===> N  (Y/N)
  C/ZOS                 ===> N  (Y/N)


          Press ENTER to process  or  enter END command to terminate
```

Note: On the Compile Information Table screen shown in the Figure 8-2 on page 8-6, enter **YES** in the **5.1 DDNames** field if you are using CA OPTIMIZER and the release 5.1 DDNames SYSUT7, SYSUT8, and CAISTATS are to be allocated.

2. Enter Compiler Libraries

   This option allows you to supply the library names for the compiler you selected in Option 1. See Figure 8-4 on page 8-7 through Figure 8-7 on page 8-8. The compilers that you selected will have their corresponding arrow indicators highlighted and the field names will be modifiable.

Figure 8-4. Compile Information Table - Enter Compiler Libraries (1 of 4)

```
Compuware CSS Utilities ------ Compiler Libraries ------------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:     +
COBOL Libraries:
  OS/VS COBOL          ===>
  Linkedit SYSLIB      ===>

  VS COBOL II          ===>
  Linkedit SYSLIB      ===>

  COBOL/370            ===>
  Linkedit SYSLIB      ===>

  COBOL/MVS            ===>
  Linkedit SYSLIB      ===>

  COBOL/390            ===>
  Linkedit SYSLIB      ===>

  ENTERPRISE/COBOL     ===>
  Linkedit SYSLIB      ===>

          Press ENTER to process  or  enter END command to terminate
```

Figure 8-5. Compile Information Table - Enter Compiler Libraries (2 of 4)

```
Compuware CSS Utilities ------ Compiler Libraries ------------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:   - +
  Linkedit SYSLIB      ===>

  CA OPTIMIZER         ===>
  Linkedit SYSLIB      ===>

ASSEMBLER Libraries:
  Assembler H          ===>

  High Level Assembler ===>

PL/I Libraries:
  PL/I                 ===>
  Linkedit SYSLIB      ===>
                       ===>

  PLI/370              ===>
                       ===>
  Linkedit SYSLIB      ===>

          Press ENTER to process  or  enter END command to terminate
```

Figure 8-6. Compile Information Table - Enter Compiler Libraries (3 of 4)

```
Compuware CSS Utilities ------ Compiler Libraries -----------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:   - +
  Linkedit SYSLIB      ===>

  VA/PLI               ===>
  Run-time LIBRARY     ===>
  Linkedit SYSLIB      ===>

  ENTERPRISE/PLI       ===>
  Run-time LIBRARY     ===>
  Linkedit SYSLIB      ===>

C Language Libraries:
  C/MVS                ===>
  LE LIBs              ===>
  Compiler SYSMSGS     ===>
  Linkedit SYSLIB      ===>

  C/390                ===>
  LE LIBs              ===>

          Press ENTER to process  or  enter END command to terminate
```

Figure 8-7. Compile Information Table - Enter Compiler Libraries (4 of 4)

```
Compuware CSS Utilities ------ Compiler Libraries -----------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                    More:   -
  Linkedit SYSLIB      ===>

C Language Libraries:
  C/MVS                ===>
  LE LIBs              ===>
  Compiler SYSMSGS     ===>
  Linkedit SYSLIB      ===>

  C/390                ===>
  LE LIBs              ===>
  Compiler SYSMSGS     ===>
  Linkedit SYSLIB      ===>

  C/ZOS                ===>
  LE LIBs              ===>
  Compiler SYSMSGS     ===>
  Linkedit SYSLIB      ===>

          Press ENTER to process  or  enter END command to terminate
```

Libraries entered on these screens depend on the compile processors used at your site. The following types of libraries can be entered on this screen:

- Compiler libraries
- SYSLIB libraries
- Compiler dependent libraries

Compiler libraries describe the datasets that contain the compile processor load modules. These libraries are required only if the processor libraries are not already included in the concatenation lists for LPA or LINKLIST. When these libraries are supplied they will appear in the STEPLIB DD of the compile JCL.

The SYSLIB libraries describe the datasets to be used for the primary resolution of external references by the linkage editor or binder. For example, the L/E library, SCEELKED. When these libraries are supplied, they will appear in the SYSLIB DD of the linkedit step.

Compiler dependent libraries are for those compilers that have additional library requirements. For example, the C language SYSMSGS library or the L/E library, SCEERUN.

3. Enter DB2 IDs and Precompile Libraries (Optional)

   If your site has DB2, this option will allow you to define DB2 identifiers and associate the identifiers with DB2 libraries. See Figure 8-8 and Figure 8-9 below. When an identifier is selected, the associated datasets will appear in the STEPLIB DD of the pre-compile step and in the SYSLIB DD of the linkedit step. The DB2 IDs will appear on the Compile Menu. See Figure 8-13 on page 8-13.

Figure 8-8. Compile Information Table - Enter DB2 IDs and Precompile Libraries (1 of 2)

```
Compuware CSS Utilities --- DB2 Precompile Libraries --------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                More:     +
  These libraries are used in the DB2 precompile step.  The ID is displayed
  on the 'Compile Menu'.  When the programmer selects one of the IDs from
  the 'Compile Menu', the ID is matched to the corresponding library name(s)
  that are listed here.  You may have up to two libraries with the same ID
  name.  The 'Compile Menu' can display up to 8 unique IDs.

            ID                DB2 Precompile Libraries
 (1) ===>        ===>
 (2) ===>        ===>
 (3) ===>        ===>
 (4) ===>        ===>
 (5) ===>        ===>
 (6) ===>        ===>
 (7) ===>        ===>
 (8) ===>        ===>
 (9) ===>        ===>
(10) ===>        ===>
(11) ===>        ===>

          Press ENTER to process  or  enter END command to terminate
```

Figure 8-9. Compile Information Table - Enter DB2 IDs and Precompile Libraries (2 of 2)

```
Compuware CSS Utilities --- DB2 Precompile Libraries --------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                More:   -

            ID                DB2 Precompile Libraries
 (1) ===>        ===>
 (2) ===>        ===>
 (3) ===>        ===>
 (4) ===>        ===>
 (5) ===>        ===>
 (6) ===>        ===>
 (7) ===>        ===>
 (8) ===>        ===>
 (9) ===>        ===>
(10) ===>        ===>
(11) ===>        ===>
(12) ===>        ===>
(13) ===>        ===>
(14) ===>        ===>
(15) ===>        ===>
(16) ===>        ===>

          Press ENTER to process  or  enter END command to terminate
```

In the column on the left, type in a DB2 identifier. It can be from one to six characters in length. You may have up to two identifiers with the same name. In addition, you may have up to eight unique DB2 identifiers.

In the column on the right, type in the DB2 dataset names that are required to perform the DB2 precompile step.

For Example:

```
           ID                DB2 Precompile Libraries
(1) ===> D610S1 ===> 'SYS1.DSN610.SDSNLOAD'
(2) ===> D610S1 ===> 'SYS1.DSN610.SDSNEXIT'
(3) ===> D710S1 ===> 'SYS1.DSN710.SDSNLOAD'
(4) ===> D710S1 ===> 'SYS1.DSN710.SDSNEXIT'
(5) ===> D810S1 ===> 'SYS1.DSN810.SDSNLOAD'
(6) ===> D810S1 ===> 'SYS1.DSN810.SDSNEXIT'
```

There are two D610S1 identifiers. That is the maximum for a single identifier. Also, there are three unique identifiers D610S1, D710S1, and D810S1. You may have up to eight of these.

4. Enter CICS IDs and Translator Libraries (Optional)

   If your site has CICS, this option will allow you to define CICS identifiers and associate the identifiers with CICS libraries. See Figure 8-10 and Figure 8-11 below. When an identifier is selected, the associated datasets will appear in the STEPLIB DD of the translator step and in the SYSLIB DD of the linkedit step. The CICS IDs will appear on the Compile Menu. See Figure 8-13 on page 8-13.

Figure 8-10. Compile Information Table - Enter CICS IDs and Translator Libraries (1 of 2)

```
Compuware CSS Utilities --- CICS Translator Libraries -------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                More:     +
  These libraries are used in the CICS translator step.  The ID is displayed
  on the 'Compile Menu'.  When the programmer selects one of the IDs from
  the 'Compile Menu', the ID is matched to the corresponding library name(s)
  that are listed here.  You may have up to two libraries with the same ID
  name.  The 'Compile Menu' can display up to 8 unique IDs.

            ID                CICS Translator Libraries
 (1) ===>        ===>
 (2) ===>        ===>
 (3) ===>        ===>
 (4) ===>        ===>
 (5) ===>        ===>
 (6) ===>        ===>
 (7) ===>        ===>
 (8) ===>        ===>
 (9) ===>        ===>
(10) ===>        ===>
(11) ===>        ===>

          Press ENTER to process  or  enter END command to terminate
```

Figure 8-11. Compile Information Table - Enter CICS IDs and Translator Libraries (2 of 2)

```
Compuware CSS Utilities --- CICS Translator Libraries -------------------------
COMMAND ===>
   Use 'Page UP' and 'Page DOWN' to scroll this screen.
                                                                  More:   -

            ID                CICS Translator Libraries
 (1) ===>        ===>
 (2) ===>        ===>
 (3) ===>        ===>
 (4) ===>        ===>
 (5) ===>        ===>
 (6) ===>        ===>
 (7) ===>        ===>
 (8) ===>        ===>
 (9) ===>        ===>
(10) ===>        ===>
(11) ===>        ===>
(12) ===>        ===>
(13) ===>        ===>
(14) ===>        ===>
(15) ===>        ===>
(16) ===>        ===>

          Press ENTER to process  or  enter END command to terminate
```

In the column on the left, type in a CICS identifier. It can be from one to six characters in length. You may have up to two identifiers with the same name. In addition, you may have up to eight unique CICS identifiers.

In the column on the right, type in the CICS dataset names that are required to perform the CICS translator step.

For example:

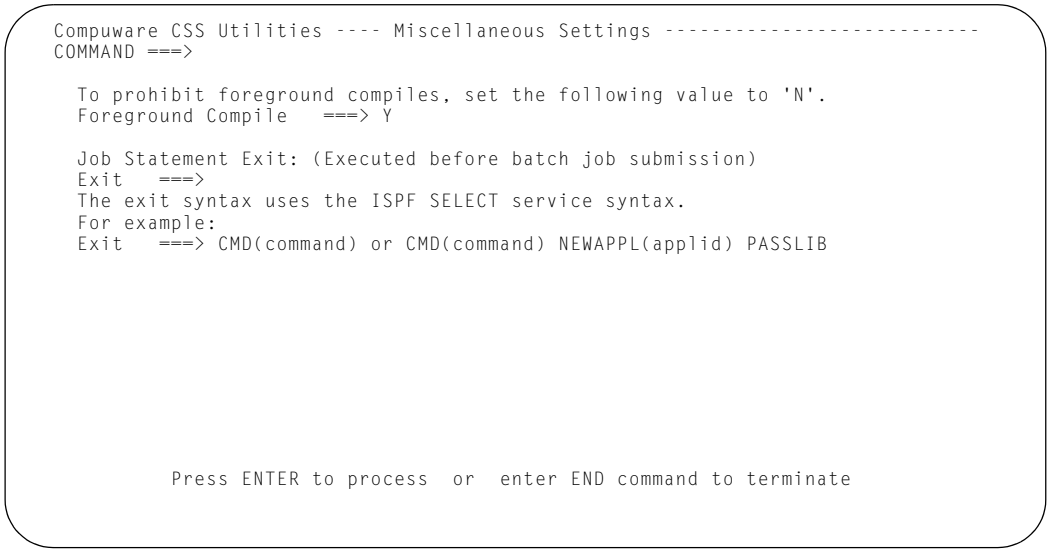
```
            ID                CICS Translator Libraries
(1) ===> CTS310 ===> 'SYS1.CICS310.LOADLIB1'
(2) ===> CTS310 ===> 'SYS1.CICS310.LOADLIB2'
(3) ===> CTS220 ===> 'SYS1.CICS220.LOADLIB1'
(4) ===> CTS220 ===> 'SYS1.CICS220.LOADLIB2'
(5) ===> CTS210 ===> 'SYS1.CICS210.LOADLIB1'
(6) ===> CTS210 ===> 'SYS1.CICS210.LOADLIB2'
```

There are two CTS310 identifiers. That is the maximum for a single identifier. Also, there are three unique identifiers CTS310, CTS220, and CTS210. You may have up to eight of these.

5. Enter Miscellaneous Information

   There are some options that don't fall under the previously mentioned options. Those items are found on this screen. See Figure 8-12 on page 8-12.

Figure 8-12. Compile Information Table - Enter Miscellaneous Information



Foreground Compile: Some sites do not want their staff doing foreground compiles. If you do not want your users doing foreground compiles, then set this field to N. The Foreground literal will not be displayed as one of the Preparation options. See Figure 8-14 on page 8-14.

Job Statement Exit: This is an exit point in which the JOB card is passed to the stated entity before job submission for the Batch option. The invocation of the exit is also done before the JCL is displayed with the Editjcl option.

The invocation of the exit uses the ISPF SELECT service syntax.

The following are examples:

```
Exit ===> PANEL(panelid) or PANEL(panelid) NEWAPPL(applid) PASSLIB

Exit ===> CMD(command) or CMD(command) NEWAPPL(applid) PASSLIB

Exit ===> PGM(program-name) or PGM(program-name) PARM(parameters) or

        PGM(program-name) PARM(parameters) NEWAPPL(applid) PASSLIB
```

Note: *Batch*, *Editjcl* and *Foreground* refer to Preparation types on the Compile Menu. See Figure 8-13 on page 8-13.

A. Process Options 1 through 5

Entering this option on the command line will cause options 1 through 5 to be displayed sequentially.

C. Copy a Prior Installation's Information to a New Library

The Copy process will take an existing compile information table with member AAUTCTBL and copy the member to a new compile information table. If the new table already has member AAUTCTBL, a message window appears asking if you want to overwrite the existing member or bypass the update.

This process may be used to create a back-up of your completed compile information table. Also, it may be used to carry existing compile information forward to a new release. Thus, avoiding having to retype the compile information.

Place a **C** on the command line. In the first field, enter the dataset name that you want to contain the compile information. In the second field, enter the dataset

name of an existing compile information table. Hit enter to copy the information.

Later on during the CSS installation, you will need the dataset name of the compile information table.

The Compile Menu is displayed below to assist you in your understanding of how the values that you supply during this installation will affect the display of the Compile Menu.

Figure 8-13. Compile Facility - Compile Menu

```
Compuware CSS Utilities -------- Compile Menu ------------------- CSS V16.05
COMMAND ===>

Primary Commands: Listing (Display output) SEtup (Display general settings)

Compile Profile: DEFAULT   - DEFAULT COMPILE PROFILE
              _ Display the Compile Profile list

Source Dsname ===> 'USERID.COB.SOURCE'
 Preparation ===> BATCH       (Batch/Editjcl/Foreground)              <-Opt 5
    Language ===> E-COBOL     Select From The List Below              <-Opt 2
 (COB/COB2/COB370/COBMVS/COB390/E-COBOL/CAOPT/HASM/HLASM/PLI/PLI370)  <-Opt 1
 (VAPLI/E-PLI/CMVS/C390/CZOS)

SEL Options:
 D - Display settings        S - Select the DB2 Precompile libraries
 S - Process only                and/or the CICS Translator libraries
 SEL        STEPS              DB2 Precompile           CICS Translator
 -- -------------------     --------------------     ---------------------
 _  1. DB2 Precompile       _ D810S1 _ D710S1     _ CTS31A   _ CTS31B <-Opt 3&4
 _  2. CICS Translator      _ D610S1 _ D810S2     _ CTS31C   _ CTS21A
 _  3. Compile              _ D710S2 _ D610S2     _ CTS21B   _ CTS21C
 _  4. Linkedit             _ D510S1 _ D510S2     _ CTS11A   _ CTS11B
```

Note: The *option* references refer to the option number on the Install Menu screen.

If you select all of the compilers, this is the list that is generated (option 1). The user will select the language he wants to use for his compile by placing one of the names from the list in the Language field. The name entered tells the compile facility which libraries to use during the compile (option 2).

If you used eight unique DB2 identifiers and libraries, then the DB2 Pre-compile list will be similar to this list (option 3). If you supplied less than eight unique identifiers, then they will appear in a left to right, top to bottom sequence.

If you used eight unique CICS identifiers and libraries, then the CICS Translator list will be similar to this list (option 4). If you supplied less than eight unique identifiers, then they will appear in a left to right, top to bottom sequence.

If you set the Foreground Compile value to **Y**, then the Foreground literal appears in the Preparation list (option 5). If you set the value to **N**, then the literal does not appear.

The following figure shows the Compile Menu with fewer values supplied during installation. In addition, the Foreground Compile option is set to **N**.

Figure 8-14. Compile Facility - Compile Menu

```
Compuware CSS Utilities -------- Compile Menu -------------------- CSS V16.05
COMMAND ===>

Primary Commands: Listing (Display output) SEtup (Display general settings)

Compile Profile: DEFAULT   - DEFAULT COMPILE PROFILE
               _ Display the Compile Profile list

Source Dsname ===> 'USERID.COB.SOURCE'
  Preparation ===> BATCH       (Batch/Editjcl)
     Language ===> E-COBOL     Select From The List Below
  (COB390/E-COBOL/CAOPT/HLASM/VAPLI/E-PLI/C390/CZOS)


SEL Options:
  D - Display settings        S - Select the DB2 Precompile libraries
  S - Process only                and/or the CICS Translator libraries
  SEL        STEPS              DB2 Precompile          CICS Translator
  --- -------------------     --------------------     ---------------------
   _  1. DB2 Precompile       _ D810S1 _ D710S1     _ CTS31A   _ CTS21A
   _  2. CICS Translator      _ D610S1              _ CTS11A
   _  3. Compile
   _  4. Linkedit
```

Step 5. Activate Compuware ISPF Dialogs (Required)

You should perform this step if you plan to use CSS Utilities.

Compuware ISPF dialogs include CSS Utilities. This section covers two different ways to activate the CSS Utilities libraries—dynamically, or at TSO startup.

Dynamically allocating the libraries is discussed in Option 1 below. Allocating the libraries at TSO startup is discussed in "Option 2: Allocate the Libraries at TSO Startup" on page 8-16. For more information about CSS Utilities functionality, see the *Compuware Shared Services User/Reference Guide*.

Option 1: Dynamically Allocate Libraries when CSS Utilities are Executed

To dynamically allocate the CSS Utilities libraries when CSS Utilities are executed, you must determine how to make the REXX EXECs and CLISTs accessible.

CSS provides sample REXX EXECs and CLISTs for invoking CSS Utilities in SLCXCNTL sample library. You can allocate this library at TSO logon time, or make the CLIST or REXX EXEC accessible in a user-defined library allocated at TSO logon.

IMPORTANT
If CSS REXX EXECs or CLISTs are currently used to invoke the CSS utilities, you must replace them with the latest versions in order to use the new Abend-AID features and Abend-AID for CICS utility features. With release 8.3, the Compile Facility has requirements for a new table library and a new skeleton library. Also with release 8.3, the CSS utilities have requirements for the Abend-AID for CICS load library when use with CICS type DDIO files is anticipated.

Complete the following steps:

1. Determine whether to use REXX EXECs or CLISTs.

   Compuware recommends using REXX EXECs; you may find them easier to customize and maintain.

2. Determine whether Japanese and English language support is required, or just English language support.
3. If using REXX EXECs, choose the appropriate REXX EXEC for CSS Utilities using Table 8-2 as a guide. If using CLISTs, select one CLIST for CSS Utilities.
4. Change the library names in the REXX EXECs or CLISTs selected in Step 3 to match your site standards.

Note: Effective with CSS Release 8.1, an optional ISPSLIB can be used with On-the-Fly processing. See Appendix D in the *Compuware Shared Services User/Reference Guide* for details about On-the-Fly processing. You may use an existing skeleton library allocated at TSO logon time, or allocate one at viewer startup by updating the REXX EXECs or CLISTs selected in "Step 3. Make New CSS Load Modules Accessible to Compuware Products (Required)" on page 8-4.

Note: Effective with CSS Release 8.3, the CSS Online Utilities require the SLCXSLIB and SLCXTABL datasets to be concatenated to ISPSLIB and ISPTLIB respectively. The SLCXTABL dataset was created in "Step 4. Create the Compile Information Table (Required)" on page 8-4.

Table 8-2. REXX EXECs or CLISTs Used to Invoke CSS utilities

Use this REXX EXEC	Or this CLIST	For
CWUTREXE	CWUTCLSE	CSS Utilities, English-only language support, set up or run XPEDITER utilities
CWUTREXJ	CWUTCLSJ	CSS Utilities, English and Japanese language support, set up or run XPEDITER utilities

5. Allocate the ECC SLCXCNTL library at TSO logon time, or copy the edited REXX EXECs/CLISTs to a library allocated at TSO logon:
   - **To allocate the ECC SLCXCNTL library at TSO logon**

     Add the library to the SYSPROC concatenation in the TSO logon PROC.
   - **To make specific REXX EXECs/CLISTs accessible in a library allocated at TSO logon**

     Copy the REXX EXECs or CLISTs to an existing SYSPROC library allocated at TSO logon.
6. Users can invoke the CSS Utilities by selecting an option from an ISPF panel (recommended), or by executing a REXX EXEC or CLIST.
   - **To enable users to select an ISPF menu option**

     Add one or more of the following lines to an ISPF panel. Note that 'o' represents the option to be entered at the menu command prompt.

To invoke CSS Utilities	o,'CMD (%*CSS Utilities CLIST name* **or** *CSS Utilities REXX EXEC name*)

   - **To invoke CSS Utilities by executing a REXX EXEC or CLIST**

     Enter the appropriate command at the TSO command prompt:

To invoke CSS Utilities	TSO %*CSS Utilities CLIST name* **or** *CSS Utilities REXX EXEC name*

Option 2: Allocate the Libraries at TSO Startup

Allocating the load, panel, and message libraries at TSO logon simplifies maintenance. All TSO users have access to CSS Utilities after logging on to TSO, and upgrades that affect all TSO users can be made in one central location. Allocating these libraries at TSO logon may be inappropriate, however. In that case, dynamically allocating the libraries may be a better solution.

To provide access to the load, panel, and message modules at TSO logon time, complete the following steps.

1. Concatenate each library as shown in Table 8-3:

Table 8-3. Allocating ECC Libraries

Load library SLCXLOAD	Place the ECC load library in the link list (see "Step 3. Make New CSS Load Modules Accessible to Compuware Products (Required)" on page 8-4 for instructions) OR Place a DD statement for the load library in the concatenation for STEPLIB or ISPLLIB in your TSO logon PROC OR Execute a CLIST or REXX EXEC at logon time that will concatenate the load library to the appropriate DD statement before starting ISPF **Note:** Abend-AID for CICS users must place a DD statement for the load library in the concatenation for FDBDRPL in your server startup JCL.
Message library SLCXMENU	Concatenate the library to ISPMLIB in the TSO logon PROC OR Concatenate the library to ISPMLIB in a CLIST or REXX EXEC before starting ISPF
Panel library SLCXPENU	Concatenate the library to ISPPLIB in the TSO logon PROC OR Concatenate the library to ISPPLIB in a CLIST or REXX EXEC before starting ISPF
EXEC library SLCXEXEC	Concatenate the library to SYSPROC in the TSO logon PROC OR Concatenate the library to SYSPROC in a CLIST or REXX EXEC before starting ISPF
Skeleton Library SLCXSENU	Due to the Compile Facility's requirement for this library, as of CSS release 8.3, this DD is required. Concatenate the library to ISPSLIB in the TSO logon PROC OR Concatenate the library to ISPSLIB in a CLIST or REXX EXEC before starting ISPF. Optional - concatenate the library for On-the-Fly processing. See Appendix D in the *Compuware Shared Services User/Reference Guide* for details.
Table Library SLCXTABL	Concatenate the library to ISPTLIB in the TSO logon PROC OR Concatenate the library to ISPTLIB in a CLIST or REXX EXEC before starting ISPF. **Note:** This file is created in "Step 4. Create the Compile Information Table (Required)" on page 8-4.

Notes:

a. Replace any existing CSS libraries in these concatenations.

b. The low level qualifiers are listed below their respective library names.

c. When testing a new install of Abend-AID, if the new Abend-AID load and customization libraries have not yet been placed in the link list, they must be added as explained in the instructions for the SLCXLOAD library. Doing this will prevent text merge error messages when viewing an Abend-AID report.

d. When testing a new install of Abend-AID for CICS, if the new Abend-AID for CICS load library has not yet been placed in the link list, it must be added as

explained in the instructions for the SLCXLOAD library. Doing this will prevent error messages when using the CSS Utilities with Abend-AID for CICS DDIO files.

2. Add one or more of the following lines to an ISPF panel. Note that 'o' represents the option to be entered at the ISPF menu command prompt.

To invoke CSS Utilities	o,'PGM(CWDDIUFE) NEWAPPL(AAUT)'

Step 6. Link CSS Utilities Tutorials to Your Main Tutorial Panel (Optional)

The ECC SLCXCNTL sample library contains two sample tutorial main menu panels for ISPF and ISPF/PDF:

- CXTUTOR
- CXR00003

To link the CSS Utilities tutorial panels to your tutorial main menu panel (TTUTOR or ISR00003), use the applicable member as an example.

Step 7. Prepare the DDIO Files

You must perform this step if you are a new CSS user or have never created DDIO files and now need to use them. If you are an Abend-AID user you will need to allocate a Report Shared Directory and one or more Report databases.

CSS DDIO files are used to store Abend-AID reports, transaction dump information, and source listings. The term *DDIO file* is a generic term used to refer to the datasets that store the reports and listings from Compuware Products that use CSS. This step describes the various types of DDIO files available and who needs to allocate which type.

Types of DDIO Files

There are two main categories of DDIO files: report or source listing files and report or source listing *databases* which are grouped into pools that are attached to Shared Directories. CSS uses the Shared Directory architecture to manage the available pool space.

Within these two categories of DDIO files, there are three different types of DDIO files characterized by the type of information that is stored in them: Abend-AID reports, source listings, and CICS transaction dump information.

Note: If you use the Shared Directory architecture for your DDIO files, different database types cannot be attached to the same Shared Directory (for example, you cannot attach a source listing database to a report shared directory, and vice versa). Starting with CSS release 8.3, you can now attach transaction databases and report databases to the same Shared Directory.

Note: Non-database report or source listing files cannot be attached to a Shared Directory until they have been converted using the batch utility CONVERT command in CWDDALLU, or CWAASDUT (report only) and CWDDLPUT (source listing only). See the "Batch File Utility" chapters in the *Compuware Shared Services User/Reference Guide* for more information, specifically Chapter 8 "Batch File Utility CWDDALLU" for detailed command and parameter descriptions.

1. Abend Reports:
   a. **Report Shared Directories and Databases.** This type of file is used to store abend reports created by Abend-AID. *This Report Shared Directory architecture is only supported for Abend-AID Release 9.4 and later.* Report databases are attached to a report shared directory. The report databases are managed as a POOL of space belonging to the report shared directory to which they are attached.
   b. **Report DDIO files.** This type of DDIO file is used to store Abend Reports created by Abend-AID. Prior to Abend-AID Release 9.4, this was the only type of report file available. For compatibility purposes, Abend-AID Release 9.4 can also use this type of report file. A conversion utility is available to run when the user is ready to migrate the report file to the report database format.

      Note: Abend-AID Classic Report files are not supported with Abend-AID Release 12.4 or newer.

2. Source Listings:
   a. **Source Listing Shared Directories and Databases.** This type of DDIO file is used to store compiler source listings for use by various Compuware Products. Source listing databases are attached to a source listing shared directory. The source listing databases are managed as a POOL of space belonging to the source listing shared directory to which they are attached. *The Source Listing Shared Directory architecture can now be used with all supported language compilers. However, XPEDITER users should see notes and exceptions below.*

b. **Source Listing DDIO Files.** This type of DDIO file is used to store compiler source listings. A conversion utility is available to run when the user is ready to migrate the listing file to the source listing database format.

The same source listing file, database, or source listing member can be used by multiple Compuware Products. It is not necessary to create duplicate source listing files, databases, or source listing members for each Compuware Product.

3. **Transaction Databases and Shared Directory for Abend-AID for CICS.** Transaction databases are used by Abend-AID for CICS to store CICS transaction dump information. Abend-AID for CICS catalogs both CICS transaction dumps and imported Region Dumps into the shared directory.

The following table lists the types of DDIO files created by the CSS utilities and which products use/require them:

Table 8-4. DDIO File Formats

Compuware Product	Report File	Source Listing File	Transaction Databases & Shared Dir.	Report Shared Dir. & Database	Source Listing Shared Dir.	Source Listing Database
Abend-AID	x[1] See Note	x		x	x	x
Abend-AID for CICS		x	x		x	x
Xpediter/TSO		x			x[2] See Note	x
Xpediter/IMS		x			x[2] See Note	x
Xpediter/CICS		x			See Note	x
Xpediter/Eclipse		x			See Note	x
Strobe		x			x	x

[1] Abend-AID Classic Report files are not supported with Abend-AID Release 12.4 or newer.

[2] If using Xpediter/TSO and /IMS with the C Language Processor, VisualAge PL/I, or Enterprise PL/I, and you use Long Program Names, the Source Listing Shared Directory must be used.

Refer to the "Source Listing Databases and Shared Directories" section of Chapter 2 in the *Compuware Shared Services User/Reference Guide* for more information.

File Access Methods

You may use either sequential or VSAM files for DDIO files. Shared directories must be VSAM files. If you are using VSAM files for your DDIO files, you can copy them using the IDCAMS REPRO command. You cannot, however, change any of the file attributes such as CISIZE without reallocating a DDIO/DB file. Allocation and format requirements, along with the steps for preparing the various DDIO files, will vary according to the type of file. Source listing files can be shared between all Compuware products that use CSS, while report files, transaction databases, work files, and profile files, cannot. Source listing files used by XPEDITER/CICS must be VSAM.

CAUTION:

CWDDSUTL, CWFXSDUT, CWAASDUT, CWDDLPUT, and CWDDALLU are the only utilities recommended for use with Compuware DDIO files.

File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can result in corrupting the contents of the DDIO file. Refer to "Chapter 2. Allocating and Formatting DDIO Files" in the *Compuware Shared Services User/Reference Guide* for more information.

Step 7a. Allocate and Format Abend-AID Report Shared Directories and Databases for Abend-AID Release 9.4 and Later Users

This step should only be performed by Abend-AID users who have installed, are installing, or are migrating to Abend-AID Release 9.4 or later. The purpose of this step is to allocate and format a Report Shared Directory and one or more Report Databases attached to that Shared Directory. Note that Abend-AID 9.4 is the minimum release able to use these types of DDIO file.

Typically, this step would be performed during Abend-AID release 9.4 or later installation. For more information, see the *Abend-AID Installation and Customization Guide*.

The CSS online utilities can be used to allocate and format Abend-AID report DDIO files. Refer to "Chapter 3. CSS Utilities" in the *Compuware Shared Services User/Reference Guide* for more information.

Create the Report Shared Directory

Use the sample JCL in the ECC SLCXCNTL dataset to allocate and format the Report Shared Directory (member is CXALLDAA). For additional options refer to the "Batch File Utility CWDDALLU" chapter in the *Compuware Shared Services User/Reference Guide*. The shared directory CREATE command will both ALLOCATE and FORMAT the shared directory. If the file currently exists, you must specify parameter REALLOCATE=YES or the allocate will fail with a duplicate file error.

Create the Report Databases

You must have already created the Report Shared Directory before attempting to create the Report Database. You **can** create both in the same job as long as the Report Shared Directory create is done first. The database CREATE command will both ALLOCATE and FORMAT the database. If the file currently exists, you must specify parameter REALLOCATE=YES or the allocate will fail with a duplicate file error.

VSAM Report Databases can be created using the sample JCL in the ECC SLCXCNTL dataset (member is CXALLVAA). To allocate and create a database that spans multi-volumes, use member CXDCRADB. To create a report shared directory in the same job, use member CXDCRASD.

Sequential Report Databases can be created using the sample JCL in the ECC SLCXCNTL dataset (member is CXALLBAA).

Step 7b. Allocate and Format Source Listing Shared Directories and Databases

This step may be performed by all users who use Source Listing support. However, releases prior to XPEDITER/TSO 7.3, XPEDITER/CICS 7.6, and Abend-AID for CICS 4.5 do not support Source Listing Shared Directories. Instead, individual source databases may be specified for lower levels. Source Listing Databases used by XPEDITER/CICS must be VSAM.

The CSS online utilities can be used to allocate and format source listing DDIO files. Refer to "Chapter 3. CSS Utilities" in the *Compuware Shared Services User/Reference Guide* for more information.

Create the Source Listing Shared Directory.

Use the sample JCL in the ECC SLCXCNTL dataset to allocate and format the Source Listing Shared Directory (member is CXALLDLP). For additional options refer to the "Batch File Utility CWDDALLU" chapter in the *Compuware Shared Services User/Reference Guide*. The shared directory CREATE command will both ALLOCATE and FORMAT the

shared directory. If the file currently exists, you must specify parameter REALLOCATE=YES or the allocate will fail with a duplicate file error.

Create the Source Listing Databases

You must have already created the Source Listing Shared Directory before attempting to create the Source Listing Database. You **can** create both in the same job as long as the Source Listing Shared Directory create is done first. The database CREATE command will both ALLOCATE and FORMAT the database. If the file currently exists, you must specify parameter REALLOCATE=YES or the allocate will fail with a duplicate file error.

VSAM Source Listing Databases can be created using the sample JCL in the ECC SLCXCNTL dataset (member is CXALLVLP). To allocate and create a database that spans multi-volumes, use member CXDCRLDB. To create a source shared directory in the same job, use member CXDCRLSD.

Sequential Source Listing Databases can be created using the sample JCL in the ECC SLCXCNTL dataset (member is CXALLBLP).

Step 7c. Allocate and Format Source Listing Files

You must perform this step if you did **not** perform "Step 7b. Allocate and Format Source Listing Shared Directories and Databases" **and** you do not have an existing source listing file or database. Source listing files used by XPEDITER/CICS must be VSAM.

Allocate the Source Listing Files

VSAM Source Listing files can be allocated using the sample JCL in the ECC SLCXCNTL dataset member CXALLVS. To allocate and format a source DDIO file that spans multi-volumes, use member CXDFMSL).

Sequential Source Listing files can be allocated using the sample JCL in the ECC SLCXCNTL dataset member CXALLDS.

Format the Source Listing Files

After the source listing files have been allocated, they must be formatted before they can be used. The formatting process initializes the directory structure and specifies the file options.

ECC SLCXCNTL library member CXFMTSL contains sample JCL to format a source listing file with appropriate defaults for the sample files allocated using members CXALLVS or CXALLDS. Adjust the formatting parameters to meet your DDIO file needs. Do not execute the CXFMTSL member, if you have already run the CXDFMSL job.

Step 7d. Allocate and Format Abend-AID for CICS Shared Directories and Transaction Databases

Normally this would be performed during the Abend-AID for CICS install procedure. For additional information, see the *Abend-AID for CICS Installation and Customization Guide*.

Create the Abend-AID for CICS Shared Directory (Allocate & Format)

Use the sample JCL to create the shared directory. ECC SLCXCNTL sample library member CXALLMC contains the sample JCL for creating a Abend-AID for CICS shared directory.

Create the Abend-AID for CICS Transaction Database (Allocate & Format)

You must create the shared directory before proceeding to this step. Use one of the following two sets of sample JCL to create each Abend-AID for CICS database as needed:

VSAM Abend-AID for CICS Database: ECC SLCXCNTL sample library member CXALLVSD contains sample JCL for creating a Abend-AID for CICS VSAM transaction database. To allocate and create a database that spans multi-volumes, use member CXDCRTDB. To create an Abend-AID for CICS shared directory in the same job, use member CXDCRTSD.

Sequential Abend-AID for CICS Database: ECC SLCXCNTL sample library member CXALLBSD contains sample JCL for creating a Abend-AID for CICS sequential transaction database.

Step 8. Implement the Language Processor JCL (Required)

See the sub-steps below to determine when you must perform this step.

The language processors support two methods of processing your programs: preprocessing and postprocessing. Both of these choices may involve modifications to your compile JCL.

You may use the preprocessor or post-processor (whichever is best for your site as needed.) This section discusses the benefits of the preprocessor and the post-processor and describes when to use them.

Refer to "CSS Language Processor (LP)" on page 2-6 for descriptions of preprocessing and post-processing and a summary of the benefits provided by each method.

To use the language processor, you must have previously completed installation of the ECC load library and also prepared a DDIO file. For each language you are using, you should modify the corresponding compiler procedure to provide for the type of processing you will be implementing.

When running the language processor as a preprocessor, you can specify assembler/compiler options in the CWPPRMO dataset or in the PARM statement passed to the preprocessor. The CWPPRMO is an input dataset that contains the language processor compiler options. Depending on the method used, it may be necessary to include the LANGPARM command with these options.

If you are currently using the post-processor, and you will be converting to the preprocessor, you are not required to spool your SYSPRINT to a temporary dataset.

Note: For each of the following steps, it is possible that you will receive a return code other than zero. Should this occur, note the accompanying language processor error message and refer to the *Enterprise Common Components Messages and Codes* guide for further information.

Step 8a. Modify the JCL to Run the COBOL Language Processor

You must perform this step if either of the following is true:

- You are installing XPEDITER/CICS for the first time.
- You are installing Abend-AID for the first time or you are upgrading a current Abend-AID installation with Abend-AID source support **and** you are licensed for the Compuware COBOL language processor.

You may optionally perform this step if you are installing Abend-AID for CICS or XPEDITER/TSO **and** you are licensed for the Compuware COBOL language processor. XPEDITER/TSO customers can either perform this step now in batch mode, or online as part of the XPEDITER/TSO installation procedure.

This step contains instructions for both the COBOL language preprocessor and post-processor. You should follow the instructions that apply to the type of processing you will be performing.

Preprocessor

SLCXCNTL sample library member CXCOBPRE contains sample JCL necessary to run the COBOL language preprocessor.

Modify the compile step of your COBOL JCL:

- On the EXEC statement, change the name of your compiler to CWPCMAIN and add the following to any existing COBOL compiler options:

```
LANGUAGE(compiler version)
```

Notes:

a. The LANGUAGE command may be specified in the CWPPRMO command stream rather than on the EXEC statement.

b. The preprocessor accepts VSCOBOLII, VSCOBOLIIREL3, VSCOBOLIIREL4, COBOL/370, COBOL/390, COBOL/MVS, and COBOLZ/OS as being equivalent because the program name is the same. The compiler that is actually executed depends on the library specified in the STEPLIB, JOBLIB, and/or LINKLIST concatenation.

- Add the ECC SLCXLOAD DSN to the STEPLIB or JOBLIB concatenation if it is not in the link list.
- Add the CWPDDIO DD statement to specify the name of the source listing DDIO file you will be using. The source listing DDIO file may be in standard DDIO, shared directory, or database format. Releases prior to XPEDITER/TSO 7.3, XPEDITER/CICS 7.6, and Abend-AID for CICS 4.5 do not support source listing shared directories. Instead, individual source databases may be specified for lower levels.

  Note: If the source member is to be used by XPEDITER/CICS, the source listing file specified here must be VSAM.

- Determine the language processor options you will use as input and add them to the EXEC parameter, or add a CWPPRMO DD statement that points to the appropriate options. Sample options are contained in SLCXCNTL sample library members CXLPCOBB (for batch programs) and CXLPCOBC (for CICS programs) to provide the smallest output to the source listing file (for minimizing space) and generate a printout of the COBOL enhanced listing. When using these options, a printout of the listing from the DDIO file contains only the source statements. The XREF and other portions of the listing are not written to the DDIO file. If you use the sample options, note the following:
  - If you do not have XPEDITER/TSO, remove DDIO(OUTPUT(FIND)).
  - If the program will be used by XPEDITER/CICS or Abend-AID for CICS, use the options in sample CXLPCOBC.
- The SYSPRINT file will be LRECL=133, RECFM=FBA.

Post-processor

You can use the post-processor to process compile listings that were previously stored, or you can modify your COBOL compile JCL to pass the compiler listing to the post-processor as described below.

- Check the COBOL compiler options.
- Change the COBOL compiler JCL to write the compiler listing (SYSPRINT) to a sequential file that can be used as input to the COBOL language post-processor. A sample of the necessary JCL is shown below.

```
//SYSPRINT DD UNIT=SYSDA,SPACE=(TRK,(25,20)),DCB=BLKSIZE=16093,
//            DISP=(MOD,PASS)         <===== CA-OPTIMIZER NEEDS 'MOD'
```

The above example is a temporary file. You can save the JCL as a permanent file to be reused as shown in the example below:

```
//SYSPRINT DD DSN=CX,STORED.LISTINGS(MEMBER)
```

- The SYSPRINT file will be LRECL=133, RECFM=FBA.
- Add SLCXCNTL sample library member CXCOB1 as the post-processor step following your COBOL compile step.
- Determine the language processor options you will use as input and add them to the EXEC parameter, or add a CWPPRMO DD statement that points to the appropriate options. Sample options are contained in SLCXCNTL sample library members CXLPCOBB (for batch programs) and CXLPCOBC (for CICS programs) to provide the smallest output to the source listing file (for minimizing space) and generate a printout of the COBOL listing. When using these options, a printout of the listing from the DDIO file will contain only the source statements. The XREF and other portions of the listing will not be written to the DDIO file. If you use the sample options, note the following:
  - If you do not have XPEDITER/TSO, remove DDIO(OUTPUT(FIND)).
  - If the program will be used by XPEDITER/CICS or Abend-AID for CICS, use the options in sample CXLPCOBC.

  The SLCXCNTL sample library members that are supplied for the COBOL language post-processor invoke the language processor in different ways:

Table 8-5. SLCXCNTL Sample Library Members for COBOL Language Processor

Member	Purpose
CXCOB1	Processes compiler listings through the language post-processor. Add this JCL after your COBOL compile step and before your link edit step (if one exists).
CXCOB2	Postprocesses compiler listings stored from a previous compile job. Use this JCL to process listings that have been previously compiled and stored in machine-readable format.
CXCOB99	Compiles COBOL programs and feeds the compiler output into the COBOL post-processor.

- The CWPPRTO file will be LRECL=133, RECFM=FBA.
- The CWPLOAD file must be allocated using DISP=OLD or DISP=SHR.
- The object module that is processed by the post-processor must be used as input to the linkedit step.

Step 8b. Modify the JCL to Run the PL/I Language Processor

You must perform this step if the following is true:

- You are installing XPEDITER/CICS or XPEDITER/TSO and you are licensed for the Compuware PL/I language processor.

This step contains instructions for both the PL/I language preprocessor and post-processor. You should follow the instructions that apply to the type of processing you will be performing.

Preprocessor

SLCXCNTL sample library member CXPLIPRE contains sample JCL necessary to run the PL/I language preprocessor.

Modify the compile step of your PL/I JCL:

- On the EXEC statement, change the name of your compiler to CWPPMAIN and add the following to any existing PL/I compiler options:

  ```
  LANGUAGE(compiler version)
  ```

  Refer to the *Compuware Shared Services User/Reference Guide* for a list of the compiler versions that can be specified with the LANGUAGE command. The default compiler for PL/I is the compiler currently in use on your system.

Note: The LANGUAGE command may be specified in the CWPPRMO command stream rather than on the EXEC statement.

- Add the ECC SLCXLOAD DSN to the STEPLIB or JOBLIB concatenation if it is not in the link list.
- Add the CWPDDIO DD statement to specify the name of the source listing DDIO file you will be using. The source listing DDIO file may be in standard DDIO, shared directory, or database format. Releases prior to XPEDITER/TSO 7.3, XPEDITER/CICS 7.6, and Abend-AID for CICS 4.5 do not support source listing shared directories. Instead, individual source databases may be specified for lower levels.

  Note: If the source member is to be used by XPEDITER/CICS, the source listing file specified here must be VSAM.

- Determine the language processor options you will use as input and add them to the EXEC parameter, or add a CWPPRMO DD statement that points to the appropriate options. Sample options are contained in SLCXCNTL sample library member CXLPPLI to provide the smallest output to the source listing file (for minimizing space). If you use these options, and you will be using the PL/I language preprocessor, add LANGUAGE(xxx) to the EXEC PARM or to the CWPPRMO DD statement.

See the *Compuware Shared Services User/Reference Guide* for more information on this JCL and PL/I language processor options.

Post-processor

You can use the post-processor to process compile listings that were previously stored, or you can modify your PL/I compile JCL to pass the compiler listing to the post-processor as described below.

- Check the PL/I compiler options.
- Change the PL/I compile JCL to write the compiler listing (SYSPRINT) to a sequential file for use as input to the PL/I language post-processor. A sample of the necessary JCL is shown below.

```
//SYSPRINT DD UNIT=SYSDA,SPACE=(CYL,(100,10)),DCB=BLKSIZE=19069,
//            DISP=(MOD,PASS)
```

  The above example is a temporary file. You can save the JCL as a permanent file to be reused as shown in the example below:

  //SYSPRINT DD DSN=CX,STORED.LISTINGS(MEMBER)

- The SYSPRINT file will be LRECL=133, RECFM=FBA.

Table 8-6. SLCXCNTL Sample Library Members for PL/I Language Processor

Member	Purpose
CXPLI	Processes compiler listings through the language post-processor. Add this JCL after your PL/I compile step and before your link edit step (if one exists). CXPLI can be customized to meet your site's requirements.
CXPLI2	Postprocesses compiler listings stored from a previous compile job. Use this JCL to process listings that have been previously compiled and stored in machine-readable format.

- The CWPLOAD file must be allocated using DISP=OLD or DISP=SHR.
- The object module that is processed by the post-processor must be used as input to the linkedit step.

See the *Compuware Shared Services User/Reference Guide* for more information on this JCL and PL/I language processor options.

Step 8c. Modify the JCL to Run the Assembler Language Processor

You must perform this step if you are licensed for the Assembler language processor option for any product.

You can optionally perform this step if you are installing XPEDITER/TSO **and** you are licensed for the Compuware Assembler language processor. XPEDITER/TSO customers can either perform this step now in batch mode, or online as part of the XPEDITER/TSO installation procedure.

This step contains instructions for both the Assembler language preprocessor and post-processor. You should follow the instructions that apply to the type of processing you will be performing.

SLCXCNTL sample library member CXASMPRE contains sample JCL necessary to run the Assembler language preprocessor.

Preprocessor

Modify the assembly step of your Assembler JCL:

- On the EXEC statement, change the name of your compiler to CWPAMAIN and add the following to any existing assembler options:

  ```
  LANGUAGE(assembler version)
  ```

  Refer to the *Compuware Shared Services User/Reference Guide* for a list of the assembler versions that can be specified with the LANGUAGE command. The default assembler is Assembler H.

  Note: The LANGUAGE command may be specified in the CWPPRMO command stream rather than on the EXEC statement.

- Add the ECC SLCXLOAD DSN to the STEPLIB or JOBLIB concatenation if it is not in the link list.
- Add the CWPDDIO DD statement to specify the name of the source listing DDIO file you will be using. The source listing DDIO file may be in standard DDIO, shared directory, or database format. Releases prior to XPEDITER/TSO 7.3, XPEDITER/CICS 7.6, and Abend-AID for CICS 4.5 do not support source listing shared directories. Instead, individual source databases may be specified for lower levels.

  Note: If the source member is to be used by XPEDITER/CICS, the source listing file specified here must be VSAM.

- Determine the language processor options you will use as input and add them to the EXEC parameter, or add a CWPPRMO DD statement that points to the appropriate options. Sample options are contained in SLCXCNTL sample library member CXLPASM to provide the smallest output to the source listing file (for minimizing space). When using these parameters, a printout of the listing from the DDIO file will contain only the source statements. The XREF will not be written to the DDIO file. If you use these options and you will use the Assembler language preprocessor, add LANGUAGE(xxx) to the EXEC PARM or to the CWPPRMO DD statement.
- Specify the compiler (and release) you will use by including it as a parameter of the LANGUAGE command in your EXECUTE parameter or CWPPRMO DD statement.

Refer to the *Compuware Shared Services User/Reference Guide* for more information on the LANGUAGE command, for a list of available compilers, and for more information on this JCL and language processor options.

Post-processor

You can use the post-processor to process compile listings that were previously stored, or you can modify your Assembler compile JCL to pass the compiler listing to the post-processor as described below.

- Check the assembler options.
- Change your assembler JCL to write the compiler listing (SYSPRINT) to a sequential file for use as input to the Assembler language post-processor. A sample of the necessary JCL is shown below.

```
//SYSPRINT DD UNIT=SYSDA,SPACE=(TRK,(25,20)),DCB=BLKSIZE=16093,
//            DISP=(MOD,PASS)
```

  The above example is a temporary file. You can save the JCL as a permanent file to be reused as shown in the example below:

```
//SYSPRINT DD DSN=CX,STORED.LISTINGS(MEMBER)
```

- The SYSPRINT file will be LRECL=133, RECFM=FBA.
- Add SLCXCNTL sample library member CXASM as the post-processor step following your assembler step. CXASM can be customized to meet your site's requirements.

  Note: The CWPLOAD file must be allocated using DISP=OLD or DISP=SHR.

- The object module that is processed by the post-processor must be used as input to the linkedit step.

Refer to the *Compuware Shared Services User/Reference Guide* for more information on this JCL and language processor options.

Step 8d. Modify the JCL to Run the C Language Processor

You must perform this step if you are installing XPEDITER/TSO and you are licensed for the Compuware C language processor.

This step contains instructions for both the C language preprocessor and post-processor. You should follow the instructions that apply to the type of processing you will be performing.

Preprocessor

SLCXCNTL sample library member CXCPRE contains sample JCL necessary to run the C language preprocessor.

Modify the compile step of your C JCL:

- On the EXEC statement, change the name of your compiler to CWPZMAIN and add the following to any existing C compiler options:

```
LANGUAGE(compiler version)
```

  Refer to the *Compuware Shared Services User/Reference Guide* for a list of the compiler versions that can be specified with the LANGUAGE command. The default compiler for C is the compiler currently in use on your system.

  Note: The LANGUAGE command may be specified in the CWPPRMO command stream rather than on the EXEC statement.

- Add the ECC SLCXLOAD DSN to the STEPLIB or JOBLIB concatenation if it is not in the link list.
- Add the **CWPDDIO DD** statement to specify the name of the source listing DDIO file you will be using. The source listing DDIO file may be in standard DDIO, shared directory, or database format. Releases prior to XPEDITER/TSO 7.3 do not support source listing shared directories. Instead, individual source databases may be specified for lower levels.
- Determine the language processor options you will use as input and add them to the EXEC parameter, or add a **CWPPRMO DD** statement that points to the appropriate options. Sample options are contained in SLCXCNTL sample library member CXLPC to provide the smallest output to the source listing file (for minimizing space). If you use these options, and you will be using the C language preprocessor, add LANGUAGE(xxx) to the EXEC PARM or to the **CWPPRMO DD** statement.

Refer to the *Compuware Shared Services User/Reference Guide* for more information on this JCL and C language processor options.

Post-processor

You can use the post-processor to process compile listings that were previously stored, or you can modify your C compile JCL to pass the compiler listing to the post-processor as described below.

- Check the C compiler options.
- Change the C compile JCL to write the compiler listing (SYSCPRT) to a sequential file for use as input to the C language post-processor. A sample of the necessary JCL is shown below.

```
//SYSCPRT DD UNIT=SYSDA,SPACE=(CYL,(100,10)),
//           DCB=(RECFM=VB,BLKSIZE=16096),
//           DISP=(MOD,PASS)
```

  For more information on C language processor JCL, see the "C Language Processor" chapter in the *Compuware Shared Services User/Reference Guide*.

  The above example is a temporary file. You can save the JCL as a permanent file to be reused as shown in the example below:

```
//SYSPRINT DD DSN=CX,STORED.LISTINGS(MEMBER)
```

- The SYSPRINT file will be LRECL=133, RECFM=FBA.
- Add SLCXCNTL sample library member CXC as the post-processor step following your C compile step. CXC can be customized to meet your site's requirements.

  Note: The CWPLOAD file must be allocated using DISP=OLD or DISP=SHR.

- The object module that is processed by the post-processor must be used as input to the linkedit step.

See the *Compuware Shared Services User/Reference Guide* for more information on this JCL and C language processor options.

Step 9. Continue with ECC Verification

Please continue by verifying the customization of ECC in Chapter 9, "Verification of ECC Component Customization".

Chapter 9.
Verification of ECC Component Customization

This chapter contains important information that can help you verify the installation of ECC components. Please review this chapter before beginning the actual installation process.

Verification of LMS Function

The verification of LMS uses the LAU to generate a License Verification Report. You can then use that report to verify that the procedures in Chapter 6, "LMS Customization" were performed correctly and LMS is operating properly.

1. Start the LAU by selecting the ISPF menu item that was added as part of "Step 2. Set Up License Administration Utility (LAU)" on page 6-4. The main Compuware License Management screen shown in Figure 6-8 on page 6-9 will be displayed.
2. Type **5** in the Option field and press Enter. The Reports screen (Figure 9-1) will be displayed.

Figure 9-1. Reports Screen

```
---------------------- Compuware License Management 16.05.00 -- Row 1 to 3 of 7
Command ===>                                                  SCROLL ===> PAGE
                              Report Selection
Select "D" for Detail, or "S" for Summary
                        Expire Date   Process DSN
_ License Verification
_ Current Cache Report
_ VSAM License File
_ SAM License File
_ SMF License Records
_ Product Activity
_ Activity Extract

Edit JCL . . . . N
Jobcard:
//Job Card information line 1
//Job Card information line 2
//Job Card information line 3
//Job Card information line 4

Select     Node      Description
  _        Node1     Production
  _        Node2     MVS Test1
  _        Node3     QA Test
```

3. Type S next to License Verification.
4. Type **Y** in the Edit JCL field.
5. Type the information for your jobcard in the four lines of the Jobcard field.

   Note: If you are not using all four lines of the job card, comment out the unused lines.

6. Type **S** next to the nodes on which you want the Report job to run. If no node is selected, the job is submitted without any specific routing.

Note: If you are running on only one node, do not make a selection here.

7. Press Enter. An Edit JCL screen will be displayed.
8. Make any necessary edits, then enter the END command. A Confirm Submission window will be displayed over the Reports screen.
9. Type **Y** in the Confirm Submission window and press Enter. The JCL to create your License Verification Report will be submitted.
10. Enter the END command. The main Compuware License Management screen (Figure 6-8 on page 6-9) will be displayed.
11. Type **X** in the Option field and press Enter to exit the LAU.
12. Use SDSF to examine the License Verification Report and ensure that the Compuware product being installed is recognized and enabled by LMS.

Host Communication Interface Verification (HCITEST)

Running the HCITEST job attempts to ping HCI to validate connections. `HOSTIP=` must contain the value of the home address in which the HCI is executing. `PORT=` must represent the port on which the HCI is listening.

Figure 9-2. Sample JCL in SLCXCNTL, HCITEST member

```
//*
//*  CSS TP COMMUNICATION VERIFICATION UTILITY
//*
//*  JOBSTREAM TO VERIFY THAT THE HCI AND CSS TP ARE INSTALLED
//*  AND CONFIGURED.
//*
//*
//IVP      EXEC PGM=CXTPIVP,REGION=1M,TIME=(,20)
//*
//STEPLIB  DD DISP=SHR,DSN=COMPWARE.MLCXNNN.SLCXLOAD     <== VERIFY
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
HOSTIP=MVS1.XYZCORP.COM                                  <== VERIFY
PORT=16196                                               <== VERIFY
/*
```

Figure 9-3. TP Connection established example

```
Compuware Shared Services TP Connection Verification Utility

Control statement specification:
HOSTIP=MVS1.XYZCORP.COM
PORT=16196

OPEN connection     RC = 00, ERRNO=00000
SEND data           RC = 00, ERRNO=00000
RECV data           RC = 00, ERRNO=00000
SEND data           RC = 00, ERRNO=00000
RECV data           RC = 00, ERRNO=00000

Connection to the port was successfully validated.
Valid Compuware mainframe product license verified.
```

If a connection cannot be established, or there are error conditions present, the output might resemble the example in Figure 9-4, along with suggested causes and areas that might require verification.

Figure 9-4. Example of TP Connection not established

```
Compuware Shared Services TP Connection Verification Utility

Control statement specification:
HOSTIP=MVS1.XYZCORP.COM
PORT=46806

OPEN connection     RC = 00, ERRNO=00000
SEND data           RC = 00, ERRNO=00000
RECV data           RC = 28, ERRNO=00054

The connection was unexpectedly terminated.  Possible causes:
* HCI configuration macros HCICNPCB or HCICNTPT are not correctly set.
* The PORT number specified is a secure port via AT-TLS.
* The PORT number specified is an active port, but is in use by  another kind of
application.
```

Different messages could be output depending on the exact return code received during operation.

If a connection to HCI is established, messages will be issued and will be visible in the HCI job log. The messages might resemble the example in Figure 9-5.

Figure 9-5. Example messages from an established HCI connection

```
CXTPMAI009I 0001 Init CXTP01   H08824D0
CXTPCFG019I 0001 Configuration file CXTP01   successfully processed
CXTPMAI043I 0001 Ping from 10.10.0.200
CXTPMAI007I 0001 Term RC=00000000 FDBK=0 Reason=00000000
```

Additionally, messages will be shown in the job log of the CXTPIVP utility execution. These messages might resemble the example in Figure 9-6.

Figure 9-6. Example job log messages from CXTPIVP utility execution

```
IEF403I CXTPIVP0 - STARTED - TIME=07.14.59
+CXCM001I HOST 10.10.0.200 ACTIVE
+CXCM002I ATTEMPTING CONNECTION WITH IP 10.10.0.200:16196
+CXCM003I CONNECTION SUCCESSFUL
+CXCM005I CONNECTION CLOSED
IEF404I CXTPIVP0 - ENDED - TIME=07.14.59
```

IMPORTANT: It is recommended that once the HCI is implemented, it should be set up to run as a started task and included in the IPL startup sequence.

ECC Installation, Customization and Verification Completion

The ECC installation and customization procedures are now complete. Please return to your product's installation and configuration guide and proceed with the installation instructions provided there.

Note: If you have installed the Base Services and Host Communications Interface (HCI) components, you should refer to the appropriate product manuals for specific customization/configuration instructions.

Glossary

Abend-AID. The Compuware product for fault diagnosis that provides an immediate, detailed analysis of application program failures in a comprehensive report format.

Abend-AID Fault Analytics. The Compuware Abend-AID option that assists managers in monitoring, measuring, controlling, and improving the fault resolution process in the IT environment. Abend-AID Fault Analytics (formerly Abend-AID Fault Manager) offers comprehensive fault management of the IBM OS/390, IBM z/OS, Windows 2000/NT, IBM/AIX, HP/UX, and Sun Solaris UNIX environments

Abend-AID for CICS. The Compuware product for fault diagnosis that provides a full range of analysis functions for managing abnormal CICS transaction terminations (abends) and CICS region outages.

Abend-AID for WebSphere (MQ). The Compuware product that enables developers to clearly define reason, return, and error codes, obtain MQ environment information whenever an application fails, and quickly solve enterprise and e-business systems problems. Comprehensive support is provided for WebSphere MQ/MQSeries in batch, IMS, and CICIS environments.

Abend-AID products. A generic name for the Abend-AID product family including Abend-AID (MVS), Abend-AID for CICS, Abend-AID Fault Manager, and Abend-AID for WebSphere (MQ).

Abend-AID Report. (1) A set of records containing data extracted at abend time from the affected MVS region and associated control blocks that are stored in the Abend-AID DDIO report file. (2) A readable report produced at view time using the records stored in the Abend-AID DDIO report file.

Abend time. The time when an abend occurs and Abend-AID products perform their analyses.

Allocation group. A set of blocks assigned to a DDIO file member when the member requires additional storage.

APF Authorization. APF can be used to restrict access to system functions and can require that all modules loaded by an authorized program be loaded from an authorized load library. All modules loaded by LMSINIT must be loaded from authorized load libraries or LMSINIT will fail. Only individuals with proper RACF authority will have permission to update authorized load libraries.

Automatic lock. Automatic locks are created whenever a member is added to a DDIO file format using the AUTODELETE=DUPS or AUTODELETE=STAGED option. The most recent member is automatically locked. See also **manual lock**.

AUTODELETE. An attribute of a DDIO file, specified during formatting of the file, that determines the action to be taken when the file becomes full and an attempt is made to add a new member to the file.

Base Services. Former component of Abend-AID Common Components of the Abend-AID product line. Base Services is an application framework providing batch, server, subsystem and presentation components.

Batch File Utility. A set of CSS utilities used to prepare, maintain, and manipulate DDIO files, shared directories, and databases.

CEC. Central Electronic Complex designates the physical computer which can contain one or more LPARs. CECs can be of the older type designated as G1 through G6 or can be the new z/800 or z/900.

CEC Licensing. The number of MSUs provided by that CEC across all LPARs.

CGI. Common Gateway Interface. A way of interfacing computer programs with HTTP or WWW servers, so that a server can offer interactive sites instead of just static text and images.

CGI-bin. Common Gateway Interface-BINaries. A special directory where common gateway interface (CGI) scripts are kept.

Compuware Mainframe Services Controller (CMSC). A centralized facility providing license management and common parameter library services. The CMSC also has the ability to automatically start HCI address spaces.

Compuware Shared Services (CSS). A set of components used by several Compuware products to provide storage, retrieval, and maintenance for source listings and diagnostic reports.

Compuware Installer. The Compuware Installer consists of a set of REXX EXECs, JCL skeletons, and panels. This facility prompts you for the installation information and then builds the jobs

required to perform the SMP/E installation of Compuware mainframe products.

Compuware Viewing Facility (Compuware/VF). An obsolete online ISPF application found in earlier releases of CSS. VF provided menu-driven access to DDIO files such as Abend-AID reports, databases, shared directories and source listings. Compuware/VF supported online viewing, printing, locking, unlocking and deleting of members in any DDIO file. Most batch utility functions were also available, including creating and deleting databases or shared directories. Source listings on remote files can be also be viewed. The Compuware/VF was phased out after the inception of the Abend-AID Viewer.

Customer Modification Facility (CMF). A method for activating specialized code modifications to the CSS base code. This facility can retrofit your restricted zaps from a release prior to CSS 7.4 to the SMP/E environment.

Customized Translation Table. An optional table provided by Compuware customers for translating non-printable characters. It is used by the batch file utility for printing or displaying lines containing storage displayed by Abend-AID products at abend time.

CWAASDUT. The batch file utility used for Abend-AID shared directories and report databases. CWAASDUT allows you to manage the files used by an Abend-AID shared directory and its associated abend report files. The shared directory is used to process all directory requests related to an abend report file and maintain information about database activity.

CWASSECU. The name of the user-coded Security Exit program.

CWDDALLU. The batch file utility program that can manage all source listing and report file DDIO types previously handled separately by CWAASDUT, CWDDSUTL, CWFXSDUT and CWDDLPUT.

CWDDLPUT. The batch file utility program used to manage source shared directories and databases for all of the supported languages. The shared directory maintains information about source listing processing and database activity.

CWDDSUTL. The program for the batch file utility used by Abend-AID, Xpediter/CICS, and Xpediter/TSO. CWDDSUTL allows you to print, delete, lock, unlock, copy, move, import, and export DDIO file. members, and to list DDIO file directory entries. It also allows you to initially format a DDIO file.

CWFXSDUT. The batch file utility used by Abend-AID for CICS. CWFXSDUT allows you to manage the files used by a shared directory and its associated transaction databases. The shared directory is used to process all directory requests related to a transaction database.

CWPDDIO. The DDIO file that contains the source member output generated during language processing.

CWPERRM. Dynamically allocated dataset generated during language processing containing all the CSS messages.

CWPPRMO. A language processor input dataset that contains all of the language processor options. This DD statement can also be used to specify in-stream options.

CWPPRTI. Compiler or assembler listing input to the CSS postprocessor.

CXTRACE. A report that CSS technical support requests to assist in problem resolution. This report traces processing through CSS code and can be a useful tool for determining the cause of an abend.

DBMODEL. The name of an OBJECT that contains the specifications required to create a new source listing database during dynamic database creation (DYNCREATE). The information is stored in the specified source listing shared directory. It is used during language processing to dynamically create a new source listing database to contain the source member if all attached databases are full.

DDIO (Dump Dataset Input Output). A proprietary file access method that stores Abend-AID reports, transaction dump information, and source listings within specially formatted DDIO files for several Compuware products that use CSS. You can allocate DDIO files as VSAM or sequential datasets depending on specific product requirements.

DDIOCALC. An online calculator provided to calculate optimal allocation and formatting parameters for DDIO files. DDIOCALC cannot be used for shared directories.

DDIO File. A generic term that refers to the DDIO database or dataset used to store Abend-AID reports, transaction dump information, or source listings from Compuware products that use CSS.

DDIO File Member. A generic name for an individual diagnostic report in an Abend-AID report file, a source listing in a source listing file, or transaction dump information in a transaction database.

Directory entry. A record in the fixed portion of the DDIO file (Directory) that contains information specific to a member. Each member has a corresponding directory entry. The number of directory entries is specified during formatting of the DDIO file and cannot be changed without reformatting the file.

DIRX Report. A report produced by the DIRX command of the batch file utility or the "I" line command of the Compuware V/F utility panel. This report contains attribute and alloaction information about the specified DDIO file and a listing of the directory entries.

DYNCREATE. A term used to denote the "dynamic database creation" process that can be invoked when using the CSS Language Processor with a source shared directory listed for the compiler output (CWPDDIO). DYNCREATE defaults to YES, and does not need to be declared. DYNCREATE can be turned off at create time of the DBMODEL with DYNCREATE=NO or with the change command using the same parameter declaration. A DBMODEL is defined in the desired source shared directory using the batch utility CWDDLPUT or CWDDALLU (with DYNCREATE=YES), which allows invoking of the dynamic database creation process. When there is not enough space in any of the attached source listing databases, then the DYNCREATE process is invoked. A new source listing database is created and attached, and the new source member is written.

Enhanced Listing. A convenient source of quick reference information and program documentation that merges DMAP and CLIST information, in addition to error and diagnostic messages, with a COBOL source listing.

Enterprise Common Components (ECC). ECC is the single install image for the following Compuware facilities:

- Compuware Mainframe Services Controller (CMSC)
- Compuware Shared Services (CSS)
- License Management System (LMS)
- Host Communications Interface (HCI)
- Base Services

Entry. A generic name for a transaction diagnostic report in a transaction database for Abend-AID for CICS.

Formatting a DDIO File. The preparation of an allocated VSAM or sequential file to be used as a DDIO file.

Full DDIO File. A DDIO file for which all allocation groups and/or all directory entries are allocated to members. This scenario invokes the AUTODELETE process.

Host Communications Interface (HCI). A facility that provides connectivity between mainframe-based programmer productivity software and peer node software running on other platforms in a network. It allows application programs to communicate over any one of several protocols without knowledge of which protocol is in use at any given time.

Language Processor (LP). The CSS processor that analyzes and builds Assembler, C, COBOL, or PL/I compiler output into sort work records. The C and COBOL processors then sort and merge these records with the listing. The Assembler and PL/I processors build an incore symbol table of all the identifiers. Processor control blocks are then generated and written to a source listing DDIO file. These DDIO members can then be used as input for Compuware products that require this information.

License Administration Utility (LAU). The LAU is the license administration control center for your organization's IT professional who is responsible for managing your access to Compuware products. The LAU is an ISPF application that enables

- the creation of a License File
- the import of License Certificates into a License File
- the maintenance and export of a License File
- the reporting and analysis of your License File and License Certificate information in virtual storage

Your organization's License Administrator will also set up License Management System parameters and system operation options using the LAU.

License Certificate. English-like readable electronic records that contain a portion of the information from your license agreement for a product release's use at a particular site. The License Certificate is used to update your License File.

License File. Dataset containing imported License Certificate information for all licensed releases of Compuware products. This file is in ASCII text format.

License Management System (LMS). Facility that enables you to centrally administer Compuware product License Certificates and manage access to Compuware products at your site. The LMS includes several components that together enable you to establish, maintain, diagnose, and upgrade access to the Compuware products licensed by your enterprise. The LMS replaces the

customer profile utility provided with earlier versions of Compuware mainframe products.

Line command. A command entered next to the line to be processed. A line command is executed only for the specified member.

LMVERIFY. See **Verification Report**.

Locked member. A DDIO file member that was manually locked using the batch file utility or automatically locked as a result of AUTODELETE=DUPS or AUTODELETE=STAGED processing. A manually locked member is identified by an M in the directory report or directory screen while an automatically locked member is identified by an L in the directory report or directory screen. When manually locked, a member cannot be automatically deleted when the DDIO file becomes full.

LPAR. Logical Partition is one operating system image, such as OS/390, z/OS, or z/OS-e.

LPAR Licensing. Refers to the defined capacity or "capped capacity" of the LPAR itself.

Manual lock. A member that was locked by using the LOCK batch command. Members that have been manually locked retain their locks during the autodeletion process. See also **automatic lock**.

MIPS. Millions of Instructions per Second. A unit of measure of processing performance equal to one million instructions per second.

MSU. Millions of Service Units per Hour.

Oldest member. (1) For report files, the diagnostic report with the earliest abend date and time. (2) For source listing files, the program with the earliest compile date and time.

Primary Command. A command entered in the COMMAND INPUT field while in TSO.

Problem Documentation Utility. A CSS utility that captures the documentation needed by Compuware for resolving technical issues and problems. This utility allows you to collect the documentation — such as copybooks, SYSPRINT, and DDIO members — that CSS Technical Support requires for problem resolution.

Report File. An Abend-AID DDIO file containing diagnostic reports.

Security Exit Program. A user-coded program that allows or denies execution of a requested command for a selected member by a specific user against particular DDIO files.

Server Grace. The amount of time an LPAR can run without being connected to the server.

Shared Directory. A variable-length record VSAM RRDS that maintains information about abends and language processing along with the attached database activity. A shared directory can contain Abend-AID for CICS directory records for each region and transaction dump known to a server, Abend-AID directory records for abend report processing, or source listing shared directory records necessary to process source listing database members.

SMP/E. System Modification Program/Extended is IBM's standard facility for installing and maintaining software modifications in the MVS environment.

Source Listing. A compiled or enhanced listing and additional information about a program, that is stored in a source listing DDIO file.

Source Listing Database. A specially formatted source listing DDIO file, attached to and managed by a source shared directory.

Source Listing File. A DDIO file containing source listings generated by the Language Processor.

STROBE. The STROBE MVS Application Performance Measurement System is a product that determines where and how application time is spent in online regions and batch processing programs and how system resources are used. STROBE collects several types of data as it tracks activity within an MVS environment and produces a collection of reports that helps you determine where to revise applications to improve their performance.

Suppressed statements. Statements that are not displayed in the compiler listing. Suppressed statements in a DDIO source member contain an internal flag that the Xpediter/TSO product can use to display or suppress source statements using the GEN command.

Transaction Database. A DDIO file, containing transaction dump information, that is managed by a shared directory in Abend-AID for CICS. Multiple entries created from CICS transaction terminations are stored in a single transaction database.

Unlocked Member. A DDIO file member that is not currently locked. The locked member can be unlocked either manually, using the batch file utility or automatically as a result of AUTODELETE=DUPS or AUTODELETE=STAGED processing. An unlocked member in a full DDIO file formatted with AUTODELETE=YES, AUTODE-

LETE=DUPS, or AUTODELETE=STAGED can be automatically deleted when another member is added to the file.

Verification Report (LMVERIFY). Displays the result of a verification program that checks out each product version in cache. This data is exactly what a product would receive when it runs. Running this report will prove that the License Management System environment is properly set up for the customer.

View time. The time when a diagnostic report or source listing is presented in a readable format for online viewing or printing.

.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

X