

CA-IDMS[®]

Usage Under BS2000/OSD

15.0



Computer Associates™

This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

This documentation may not be copied, transferred, reproduced, disclosed or duplicated, in whole or in part, without the prior written consent of CA. This documentation is proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of this documentation for their own internal use, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the confidentiality provisions of the license for the software are permitted to have access to such copies.

This right to print copies is limited to the period during which the license for the product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to return to CA the reproduced copies or to certify to CA that same have been destroyed.

To the extent permitted by applicable law, CA provides this documentation "as is" without warranty of any kind, including without limitation, any implied warranties of merchantability, fitness for a particular purpose or noninfringement. In no event will CA be liable to the end user or any third party for any loss or damage, direct or indirect, from the use of this documentation, including without limitation, lost profits, business interruption, goodwill, or lost data, even if CA is expressly advised of such loss or damage.

The use of any product referenced in this documentation and this documentation is governed by the end user's applicable license agreement.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227-7013(c)(1)(ii) or applicable successor provisions.

First Edition, October 2001

© 2001 Computer Associates International, Inc.
All rights reserved.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Contents

About This Guide	xi
Chapter 1. BS2000/OSD Environment	1-1
1.1 BS2000/OSD Product Requirement List	1-3
1.2 Defining the CA-IDMS Environment in BS2000/OSD	1-4
1.2.1 Message File	1-4
1.2.2 Defining the CA-IDMS user ID in BS2000	1-5
1.3 Library Organization within CA-IDMS	1-6
Chapter 2. Utilities	2-1
2.1 JCL Procedures	2-3
2.1.1 AJNL	2-3
2.1.2 BCF	2-3
2.1.3 COMP	2-3
2.1.4 CA-Culprit	2-4
2.1.5 DC	2-4
2.1.6 DDDL	2-4
2.1.7 FILE	2-4
2.1.8 GSPIX	2-5
2.1.9 IDMSMOD: Customization and Modification Utility	2-5
2.1.9.1 ADSOMAIN	2-5
2.1.9.2 APPLYAPA	2-5
2.1.9.3 BTCIDXIT	2-5
2.1.9.4 CREASYSI	2-6
2.1.9.5 CULXPROF	2-6
2.1.9.6 DCDM	2-6
2.1.9.7 DCPARM	2-6
2.1.9.8 EDTENVIR	2-6
2.1.9.9 GENSERV	2-6
2.1.9.10 GSPIEXIT	2-6
2.1.9.11 HELP	2-6
2.1.9.12 IDMSDMLC	2-6
2.1.9.13 IDMSOPTI	2-7
2.1.9.14 PMOPT	2-7
2.1.9.15 PREPSUBS	2-7
2.1.9.16 REMOVAPA	2-7
2.1.9.17 RHDCCTAB	2-7
2.1.9.18 RHDCOPTF	2-7
2.1.9.19 RHDCSRTT	2-7
2.1.9.20 RHDCUXIT	2-7
2.1.9.21 SETDEFLT	2-7
2.1.9.22 STGIDCMD	2-8
2.1.9.23 SVCOPT	2-8
2.1.9.24 USRIDXIT	2-8
2.1.9.25 WTOEXIT	2-8
2.1.9.26 WTOREXIT	2-8
2.1.10 OLQBATCH	2-8

2.1.11 PLUS	2-9
2.1.12 PRLG	2-9
2.2 Notes for the Database Administrator	2-10
2.2.1 AJNL and PRLG ENTER Files	2-10
2.2.2 GSPI Enter File	2-10
2.2.3 APARs	2-10
APAR Format	2-11
Applying Optional APARs	2-11
2.3 IDMS Utility Interface with BS2000/OSD	2-12
System File Assignment Example	2-12
2.4 BS2KOBJM Utility	2-16
Input Module	2-16
Output Module	2-16
Run Termination	2-16
2.5 Tapes and Sequential Files	2-17
Chapter 3. Address Space Layout	3-1
3.1 Introduction	3-3
3.2 Startup Processing	3-4
3.2.1 Startup Phase 1	3-4
3.2.2 Startup Phase 2	3-5
3.3 Startup Parameters	3-7
Information Returned by the Startup	3-7
Defining the REGION Parameter	3-7
3.3.1 Example	3-8
Chapter 4. CA-IDMS/DB	4-1
4.1 Alignment Requirements	4-3
4.2 COBOL Database Procedures	4-4
4.3 SYSIDMS Support	4-5
4.4 Interface for Application Programs	4-6
4.4.1 Primary Interface Module	4-6
How it Works	4-6
IDMSOPTI Module	4-6
IDMSCANC Module	4-6
COMP Procedure	4-6
4.4.2 Streamline Batch to CV Interface	4-7
Streamline Batch to CV Interface	4-7
IDMSOPTI Module	4-7
IDMSCANC Module	4-7
IDMSMOD Function BTCIDXIT	4-7
4.5 QSAM Facility	4-9
Chapter 5. CA-IDMS/DC	5-1
5.1 Program Function Keys (PF-Keys)	5-3
Terminology	5-3
PF-KEYS	5-3
DCKEYS	5-4
BS2KDKEY	5-5
Explanation of Fields	5-6
5.2 Special Considerations Regarding TRANSDATA Devices	5-7

Basic Mode in DC	5-7
5.3 UCF Considerations	5-9
5.3.1 Generalities	5-9
5.3.2 UCF-BATCH	5-9
RHDCUBAT	5-9
RHDCUCFB	5-9
5.3.3 UCF-TIAM	5-10
Dynamic Logmode Selection	5-10
5.3.4 UCF-UTM	5-10
5.4 Program Loading	5-11
5.4.1 Library Search Chain	5-11
5.4.2 Notes for Central Version	5-11
5.4.3 Dynamic Subsystem Support	5-12
5.4.3.1 What is it?	5-12
5.4.3.2 Why Use Subsystems?	5-12
5.4.3.3 When Not to Use Subsystems?	5-12
5.4.3.4 Installing CA-IDMS Subsystems	5-13
5.4.3.5 Which Modules Can be Placed in a Subsystem?	5-14
5.5 Recovery from Program Errors	5-15
5.5.1 Trapping of Termination SVC	5-15
5.5.2 Recovery from Alignment Errors	5-15
5.6 Storage Protection	5-16
5.6.1 Description	5-16
5.6.2 Performance Considerations	5-16
5.6.3 FETCH Protection	5-16
5.6.4 Installation on BS2000/OSD OSD V2.0	5-16
5.6.5 Installation on BS2000/OSD OSD V3.0	5-16
5.7 STGID Task	5-18
5.8 XS Support	5-20
Chapter 6. Data Communication Device Management (DCDM)	6-1
6.1 DCDM Introduction	6-3
6.1.1 Presentation Layer	6-3
6.1.2 Session Layer	6-3
Using the Generalized SPOOL Interface with DC	6-4
OMNIS Support	6-4
6.1.3 Transport Layer	6-5
6.2 DCDM Tables	6-6
6.2.1 The Real Table of Devices	6-7
How Does DCDM Search in this Table?	6-7
6.2.2 The Logmode Table	6-7
6.2.2.1 #BS2LGM Macro Parameters	6-8
6.2.2.2 PRESENT Parameter Options	6-9
6.2.2.3 PRESOPT Parameter Options	6-10
PRESENT=975x	6-10
PRESENT=327x/975x/ADDS	6-11
PRESENT=900x	6-11
6.2.2.4 TPROPT Parameter Options	6-11
6.2.3 The PF-Key Table(s)	6-12
6.2.4 The VTSU Control Block Table	6-12

Chapter 7. Generalized Spool Interface	7-1
7.1 Introduction	7-3
7.2 Concepts	7-4
7.3 Spool Interface Control Block (SPC)	7-5
7.4 Exit Routine for BS2KGSPI	7-6
Exit Program Requirements	7-6
Chapter 8. CA-IDMS/DC-DCAM Line Driver	8-1
8.1 Data Flow Control	8-3
8.2 Implementation of Connection Proposals	8-4
Example PDN	8-4
Reaction of DCAM Driver	8-4
8.3 Passwords in DCAM	8-5
8.4 Transport Acknowledgments	8-6
8.5 Shareable DCAM Applications	8-7
8.6 Connection Retry for Printers	8-8
8.7 Support of ISO-DCAM	8-9
Chapter 9. APPC in the BS2000/OSD environment	9-1
9.1 Overview of APPC	9-3
9.1.1 Terminology	9-3
Session VS Conversation	9-4
9.1.2 How it Works	9-4
9.1.3 LU6.2 Verbs	9-5
9.1.4 Example of Conversation	9-5
Records Definition	9-5
Allocator Dialog	9-6
Allocatee Dialog	9-7
9.1.5 CA-IDMS and APPC	9-7
Scenario 1: Emulated APPC through CA-ADS, within the same central version	9-7
Scenario 2: Real APPC through CA-ADS	9-7
Scenario 3: Real APPC through PCs	9-8
9.2 SEM62 Implementation	9-9
9.2.1 Overview	9-9
9.2.2 Different Kinds of Sessions	9-9
9.2.3 Multiple Session Management	9-10
Different MODE Names	9-10
9.3 SEM62 Generation	9-11
9.3.1 Device Table	9-11
9.3.2 System Generation	9-12
CNOS and D0FI Line Definition	9-12
SEM62 LINE Definition	9-12
SEM62 PTERM Definition	9-13
9.3.3 LU-name Identification	9-13
Static Identification	9-13
Example of Configuration Defined Statically	9-14
Sysgen Compiler Input	9-15
Dynamic Identification	9-15
Example of Configuration Defined Dynamically	9-17
Sysgen Compiler Input	9-18

9.4 DCMT LU Command	9-19
Chapter 10. BS2000/OSD Services	10-1
10.1 Introduction	10-3
10.1.1 Overview	10-3
10.1.2 Terminology	10-4
SERVICE	10-4
ENTRY	10-4
10.1.3 Assumptions and Restrictions	10-5
10.2 Installation	10-6
10.2.1 Contents of the Product	10-6
10.3 Making Use of BS2000/OSD Services	10-7
10.3.1 Step 1 — Define the Service	10-7
SERVICE-NAME=	10-7
MAXTSN=	10-7
MAXABND=	10-7
ENTER=YES/NO	10-7
SRVEP=	10-8
10.3.2 Step 2 — Generate the Service	10-8
10.3.3 Step 3 — Link Your Application Program(s)	10-9
10.3.4 Step 4 — Create an ENTER File for the Service	10-9
10.3.5 Step 5 — Define the Service to CA-IDMS/DC	10-9
10.4 Exits to Program BS2KSRVB	10-10
10.4.1 SRVSTART	10-10
10.4.2 SRVPREP	10-10
10.4.3 SRVPOSTP	10-10
10.4.4 SRVSTOP	10-10
10.4.5 SRVABEND	10-10
10.5 Administration Program BS2KSRVA	10-11
10.5.1 Functionality	10-11
10.5.2 Syntax	10-11
10.6 Abend Codes	10-12
10.7 Example: Service for DMS Access	10-13
Chapter 11. CA-IDMS/Server	11-1
11.1 Introduction	11-3
11.2 CAICCI BS2000/OSD Operation	11-4
11.2.1 System Requirements	11-4
11.2.2 Description	11-4
11.3 CAICCI Task	11-5
11.3.1 CAICCI Input Parameters	11-6
11.3.1.1 Input Parameter Descriptions	11-6
CCIMPNAM=	11-6
CCIMPADR=	11-6
CCIMPSIZ=	11-7
CCI#TSN=	11-7
MMBLKSIZ=	11-7
MEMRETRY=	11-7
MSGID=	11-8
SYSID=	11-8

CCITRACE=	11-8
11.3.2 CAICCI Control Commands	11-8
Job Variable Usage	11-9
Job Variable Example	11-9
11.3.2.1 CAICCI Command Descriptions	11-10
CCI HELP	11-10
CCI CCITRACE ON / OFF	11-10
CCI RTIMER	11-10
CCI CHECK	11-10
CCI CMD	11-10
CCI DISPLAY STORAGE	11-10
CCI DISPLAY ALL USERS	11-10
CCI SHUTDOWN	11-10
11.4 CCITCP Task	11-11
SERVICE File	11-11
11.4.1 CCITCP Input Parameters	11-12
11.4.1.1 CCITCP Input Parameter Descriptions	11-12
CCIMPNAM=	11-12
TCP#PORT#=TCP/	11-12
TCP#SOCK=	11-13
MSGID=	11-13
CCITRACE=	11-13
DRVTRACE=	11-13
SOCTRACE=	11-13
MULTICON=	11-14
11.4.2 CCITCP Control Commands	11-14
Job Variable Usage	11-14
Job Variable Example	11-14
11.4.2.1 CCITCP Command Description	11-15
CCITCP HELP	11-15
CCITCP CCITRACE ON / OFF	11-15
CCITCP DRVTRACE ON / OFF	11-15
CCITCP SOCTRACE ON / OFF	11-15
CCITCP CMD	11-16
CCITCP DISPLAY ALL USERS	11-16
CCITCP DISPLAY ALL SOCKETS	11-16
CCITCP SHUTDOWN	11-16
11.5 IDMS Task	11-17
11.5.1 BS2KSTAR Input Parameters	11-17
Special Notes	11-17

Appendix A. BS2000/OSD Specific Modules	A-1
A.1 System Modules Specific for BS2000/OSD	A-3
A.2 CA-Culprit Modules Specific for BS2000/OSD	A-5

Appendix B. Error Messages and Abend Codes	B-1
B.1 Error Messages from the DCAM Driver	B-3
B.2 Error Messages from SEM62	B-5
B.3 Return Codes from BS2KDSCV	B-6
B.4 Return Codes from the Library Interface	B-7
B.5 Abend Codes from BS2KSRVD	B-9

B.6 Abend Codes	B-10
Index	X-1

About This Guide

What This Manual Describes

This manual describes the specific requirements and environment for using CA-IDMS under the BS2000/OSD operating system.

Release 15.0 Enhancements and Compatibility Issues

Enhancements

These enhancements are in addition to the CA-IDMS features mentioned in the *CA-IDMS Features Guide — Release 15.0*.

- Support for load modules created with BINDER (LLMs). CA-IDMS supports LMS R-type (load module created by TSOSLNK) as well as L-type (load module created by BINDER) load modules. See 5.4, “Program Loading” on page 5-11 for more information.
- Support for the Euro sign (€). To accomplish this, the DCAM driver supports specification of VTSU control blocks in the configuration of DCDM. The mechanism implements a complete control of VTSU, including coded character set names. Refer to Chapter 6, “Data Communication Device Management (DCDM)” for more information.

Compatibility Issues

- All JCL procedures are delivered in SDF format.
- IDMSMOD functions CO85RESO, GSPI and PREPRESO are obsolete.
- All CA-IDMS load modules are delivered as LMS L-type members, with one exception, IDMSSBS2.
- The CDMSPAM file is only used by central version; it is no longer used by other jobs.
- The COMP procedure function DMLC no longer supports the COB1 compiler.
- Subsystem support has been altered to load a private copy of the subsystem load module if the subsystem is not active. Refer to 5.4, “Program Loading” on page 5-11 for more information.

Understanding Syntax Diagrams

Look at the list of notation conventions below to see how syntax is presented in this guide. The example following the list shows how the conventions are used.

UPPERCASE OR SPECIAL CHARACTERS	Represents a required keyword, partial keyword, character, or symbol that must be entered completely as shown.
lowercase	Represents an optional keyword or partial keyword that, if used, must be entered completely as shown.
<u>underlined lowercase</u>	Represents a value that you supply.
←	Points to the default in a list of choices.
lowercase bold	Represents a portion of the syntax shown in greater detail at the end of the syntax or elsewhere in the document.
▶▶	Shows the beginning of a complete piece of syntax.
◀◀	Shows the end of a complete piece of syntax.
▶	Shows that the syntax continues on the next line.
▶	Shows that the syntax continues on this line.
▶	Shows that the parameter continues on the next line.
▶	Shows that a parameter continues on this line.
▶ parameter ▶	Shows a required parameter.
▶ parameter parameter ▶	Shows a choice of required parameters. You must select one.
▶ parameter ▶	Shows an optional parameter.
▶ parameter parameter ▶	Shows a choice of optional parameters. Select one or none.
▶ parameter ▶	Shows that you can repeat the parameter or specify more than one parameter.
▶ parameter , parameter ▶	Shows that you must enter a comma between repetitions of the parameter.

Chapter 1. BS2000/OSD Environment

- 1.1 BS2000/OSD Product Requirement List 1-3
- 1.2 Defining the CA-IDMS Environment in BS2000/OSD 1-4
 - 1.2.1 Message File 1-4
 - 1.2.2 Defining the CA-IDMS user ID in BS2000 1-5
- 1.3 Library Organization within CA-IDMS 1-6

1.1 BS2000/OSD Product Requirement List

The following table lists the different CA-IDMS products with their BS2000/OSD prerequisite products.

CA-IDMS products	BS2000/OSD prerequisite products
Any CA-IDMS product	BS2000/OSD V2.0 or higher ASSEMBLER H V1.2 or higher LMS V3.2 or higher TIAM V11.2 or higher EDT V16.6 or higher
CA-IDMS/DB CA-Culprit CA-OLQ (batch)	SORT V7.5 or higher
CA-IDMS DC/UCF	COBOL85 COBOL Compiler V2.2 or higher
CA-IDMS/DC	DCAM V13.0 or higher

1.2 Defining the CA-IDMS Environment in BS2000/OSD

1.2.1 Message File

CA-IDMS uses the MSG7 macro to write messages to the console and to the SYSOUT system file. At least one message number must be defined in the message file: IDMS000.

The user may define its own message number which is used by the central version during the display of console messages only. This message number is an input parameter to the BS2KSTAR startup module. For more information on the BS2KSTAR input parameters, see the section on *System startup under BS2000* in the *CA-IDMS System Operations* manual.

The IDMS000 message number, and optionally the user's own message number must contain only one insert parameter. An CA-IDMS message file can be created with the MSGMAKER utility and activated with the MODIFY-MSG-FILE-ASSIGNMENT command. For example:

```

/BEGIN-PROC LOG=*ALL
/ASSIGN-SYSOUT TO=idms-sysout-file
/ASSIGN-SYSLST TO=idms-syslst-file
/START-MSGMAKER
//OPEN-MSG-FILE FILE-NAME=$uid.sysmes.idms, -
// MODE=*CREATE(TYPE=*CUSTOMER, -
// PRODUCT=IDMS)
//ADD-MSG MSG-ID=IDMS000, -
// DESTINATION=*USER-TASK, -
// LANGUAGES=E(MSG-TEXT='(&00)')
//ADD-MSG MSG-ID=yourmsg, -
// DESTINATION=*USER-TASK, -
// LANGUAGES=E(MSG-TEXT='(&00)')
//END
/EXIT-PROC

```

Parameter	Purpose
<u>idms-sysout-file</u>	SYSOUT filename
<u>idms-syslst-file</u>	SYSLST filename
<u>\$uid.sysmes.idms</u>	Message filename
<u>yourmsg</u>	Message id you can assign for your specific needs

JCL Notes:

- To activate the CA-IDMS message file automatically at BS2000/OSD startup, add this line to the MIP SYSPAR file (\$TSOS.SYSPAR.MIP.nnn):

```
MSGFILE=$uid.sysmes.idms
```

2. The only required MSG-ID is IDMS000. Defining additional messages can assist in differentiating messages for multiple releases of CA-IDMS or multiple central versions.

1.2.2 Defining the CA-IDMS user ID in BS2000

For performance reasons, the CA-IDMS user ID should be run from an account containing these characteristics:

- Attribute MAX-ALLOWED-CATEGORY=*TP - This allows the system to run as a TP task.
- Privilege INHIBIT-DEACTIVATION=*YES - Allows the system to use the DEACT=N and DWTR=N parameters of the TINF macro during the startup processing.

If the standard WTOEXIT is used, these privileges are required for the AJNL and PRLG enter files:

- NO-CPU-LIMIT=*YES
- START-IMMEDIATE=*YES

1.3 Library Organization within CA-IDMS

The CA-IDMS install process creates multiple libraries in order to clearly separate those libraries containing common, infrequently changed members from libraries containing system specific members. This configuration makes it easier to setup and maintain new systems or environments because a large portion of the software can be shared by all the different systems.

The set of libraries containing system specific software, such as the DMCL, are called the DBA libraries; the global libraries are called CA-IDMS libraries. After installation, these libraries are populated:

- IDMS APAR library
- IDMS source library
- IDMS macro library
- IDMS load library
- DBA source library
- DBA load library

The DBA source library contains all system and customer specific sources, such as the source for WTOEXIT and RHDCPARM. It also includes JCL procedures that have parameters set to default (or system specific) values.

Similarly, the DBA load library contains load modules that are system or customer specific. For example, these load modules are stored in the DBA loadlib:

- IDMSDCB — Links in RHDCPARM and WTOEXIT. RHDCPARM and WTOEXIT are site specific modules; therefore, all modules that are linked with these modules must be store in the DBA loadlib.
- CULPO, CULL, CULE — These modules are linked with the CA-Culprit profile that is tailored for your site.

When the BS2000/OSD operating system loads a program, a library must be specified. To avoid knowing in which library a load module is located, all JCL procedures set the TASKLIB to the CA-IDMS load library and the EXEC is from the DBA load library. This way, if a module is not in the DBA load library, DLL loads the module from the CA-IDMS load library.

When the IDMS loader IDMSLBS2 (also known as BS2KLODR) dynamically loads a program, the CDMSLIB chain is used. This chain is determined by TFT entries starting with 'CDMSLIB'. The order of the CDMSLIB chain is:

```
CDMSLIB  
CDMSLIBA  
:  
CDMSLIBZ  
CDMSLIB0  
:  
CDMSLIB9
```

In this chain, the DBA load library should be placed prior to the IDMS load library.

In keeping with this concept, all members created by IDMSMOD functions, except APPLYAPA, APPLYAPB, REMOVAPA and REMOVAPB, are placed in the DBA libraries. When a function edits a source member, the source is extracted from the DBA library; if the source does not exist, it is extracted from the CA-IDMS source library. In any case, the edited source is saved in the DBA library.

APPLYAPA and REMOVAPA use the CA-IDMS APAR library and the CA-IDMS load library. For CA-IDMS Tools, APPLYAPB and REMOVAPB use the CA-IDMS Tools APAR library and the CA-IDMS Tools load library. The DBA load library is updated only when an APAR is applied to/removed from a customized module. APPLYAPB and REMOVAPB are installed in the DBA source library when you install the CA-IDMS Tools.

Chapter 2. Utilities

2.1 JCL Procedures	2-3
2.1.1 AJNL	2-3
2.1.2 BCF	2-3
2.1.3 COMP	2-3
2.1.4 CA-Culprit	2-4
2.1.5 DC	2-4
2.1.6 DDDL	2-4
2.1.7 FILE	2-4
2.1.8 GSPIX	2-5
2.1.9 IDSMOD: Customization and Modification Utility	2-5
2.1.9.1 ADSOMAIN	2-5
2.1.9.2 APPLYAPA	2-5
2.1.9.3 BTCIDXIT	2-5
2.1.9.4 CREASYSI	2-6
2.1.9.5 CULXPROF	2-6
2.1.9.6 DCDM	2-6
2.1.9.7 DCPARM	2-6
2.1.9.8 EDTENVIR	2-6
2.1.9.9 GENSERV	2-6
2.1.9.10 GSPIEXIT	2-6
2.1.9.11 HELP	2-6
2.1.9.12 IDMSDMLC	2-6
2.1.9.13 IDMSOPTI	2-7
2.1.9.14 PMOPT	2-7
2.1.9.15 PREPSUBS	2-7
2.1.9.16 REMOVAPA	2-7
2.1.9.17 RHDCCTAB	2-7
2.1.9.18 RHDCOPTF	2-7
2.1.9.19 RHDCSRTT	2-7
2.1.9.20 RHDCUXIT	2-7
2.1.9.21 SETDEFLT	2-7
2.1.9.22 STGIDCMD	2-8
2.1.9.23 SVCOPT	2-8
2.1.9.24 USRIDXIT	2-8
2.1.9.25 WTOEXIT	2-8
2.1.9.26 WTOREXIT	2-8
2.1.10 OLQBATCH	2-8
2.1.11 PLUS	2-9
2.1.12 PRLG	2-9
2.2 Notes for the Database Administrator	2-10
2.2.1 AJNL and PRLG ENTER Files	2-10
2.2.2 GSPI Enter File	2-10
2.2.3 APARs	2-10
2.3 IDMS Utility Interface with BS2000/OSD	2-12
2.4 BS2KOBJM Utility	2-16
2.5 Tapes and Sequential Files	2-17

2.1 JCL Procedures

The installation process creates JCL procedures in the DBA source library.

2.1.1 AJNL

This procedure offloads a disk journal file.

2.1.2 BCF

This procedure runs IDMSBCF with the BS2000/OSD editor as front-end.

2.1.3 COMP

This procedure executes most of the compilers and utilities. Edit the procedure for more detailed information.

Parameter	Applicable values	Functionality
FUNCTION	SCHEM DBTB DMCL SUB DMLA DMLC GEN DDDL MAP PUNCH	Schema compiler Create database name table Create DMCL module Subschema compiler Assembler precompiler COBOL precompiler Sysgen compiler DDDL compiler Mapping compiler Punch a load module
SOURCE	<u>filename</u>	Input source filename
MODULE	<u>filename</u>	DMCL / Subschema / User Program name
BATCH	BATCH DC DCBATCH	For FUNCTION DMLA only: tells the procedure how to link the program. DC is for use under CA-IDMS/DC. BATCH and DCBATCH are synonyms and can be used for all other cases.
LIST	<u>suffix</u>	Suffix of SYSLST filename

Note: The MODE parameter is still a procedure parameter, but it is not used. Its function is implemented in the FILE procedure.

2.1.4 CA-Culprit

This is the CA-Culprit execution procedure. Note that END-OF-FILE on CA-Culprit input parameters can be requested on terminal by typing a "/" in column 1.

=COPY, =MACRO and USE members can be obtained from a source library by specifying this file command:

```
/ADD-FILE-LINK CULLIB,F-NAME=source.lib
```

CA-Culprit user modules are loaded from the chain of load libraries defined in the FILE procedure. This contrasts to pre-12.0 releases where the load was done from the TASKLIB.

Note: The MODE parameter is still a procedure parameter, but it is not used. Its function is implemented in the FILE procedure.

2.1.5 DC

This procedure starts the central version, by entering a job.

2.1.6 DDDL

This procedure runs IDMSDDDL with the BS2000/OSD editor EDT as front-end.

2.1.7 FILE

This procedure issues an ADD-FILE-LINK command for each file that is used by the central version, utilities and user application programs. The ADD-FILE-LINK commands for the basic system are automatically included. The DBA must include the user database files, unless the filenames are defined in the DMCL.

The FILE procedure must be called once before running any procedure, utility, or application program. One of the parameters of the ADD-FILE-LINK procedure is MODE; possible values are:

Value	Impact
LOCAL	All ADD-FILE-LINK commands for programs executing in local mode are issued
CENTRAL	All ADD-FILE-LINK commands for programs executing in central version are issued
CV	All ADD-FILE-LINK commands for central version startup are issued
AJNL	All ADD-FILE-LINK commands for archive journal are issued

2.1.8 GSPIX

This procedure contains JCL for the generalized spool interface with its exit routine. For details refer to Chapter 7, “Generalized Spool Interface.”

2.1.9 IDMSMOD: Customization and Modification Utility

This procedure calls functions that define or modify many CA-IDMS parameters and options, or apply modifications to the CA-IDMS software.

The call to an IDMSMOD function results in the execution of a number of BS2000/OSD utility programs (EDT, LMS, BINDER) and/or CA-IDMS utility programs. Their output can be found in a file:

```
&SYSLSTP..&FUNCTION
```

Parameter	Description
&SYSLSTP	A procedure parameter (preset on: T.SYSLST)
&FUNCTION	Function name

The individual functions are implemented as procedures with identical names in the DBA source library. Direct execution of the procedures is also possible. The following IDMSMOD functions are available:

2.1.9.1 ADSOMAIN

Link built-in functions with ADSOMAIN for performance.

2.1.9.2 APPLYAPA

Apply an APAR to the CA-IDMS software. An APAR (also referred to as a PTF or a REP) modifies a CA-IDMS load module.

2.1.9.3 BTCIDXIT

BTCIDXIT is a startup security exit for streamlined batch interface. Refer to 4.4.2, “Streamline Batch to CV Interface” on page 4-7 for more details.

2.1.9.4 CREASYSI

Create a commonly usable SYSIDMS parameter file.

2.1.9.5 CULXPROF

Define the CA-Culprit profile. Refer to the CA-Culprit manuals and the CAIIJMP installation parameters for more details.

2.1.9.6 DCDM

Define the Data Communication Device Management parameters. Refer to Chapter 6, “Data Communication Device Management (DCDM)” for more details.

2.1.9.7 DCPARM

Define the CA-IDMS-DBDC startup parameters. Refer to the *CA-IDMS System Operations* manual, the CAIIJMP parameters and Chapter 3, “Address Space Layout” for more details.

2.1.9.8 EDTENVIR

Define the BS2000/OSD editor (EDT) interfacing with CA-IDMS. Refer to 2.3, “IDMS Utility Interface with BS2000/OSD” on page 2-12 for more details.

2.1.9.9 GENSERV

Define and generate BS2000/OSD services modules. Refer to 10.3, “Making Use of BS2000/OSD Services” on page 10-7 for more details.

2.1.9.10 GSPIEXIT

Define and generate a GSPI load module for use without IDA. Refer to Chapter 7, “Generalized Spool Interface” for more details.

2.1.9.11 HELP

Lists the available IDMSMOD functions.

2.1.9.12 IDMSDMLC

Define default parameter IDMSDMLC. For more information, see the *CA-IDMS DML Reference - COBOL* manual, Appendix A.

2.1.9.13 IDMSOPTI

Define and generate an IDMSOPTI module. Refer to the *CA-IDMS System Operations* manual for more details. This function is mainly available for compatibility reasons, using the SYSCTL and SYSIDMS files are more flexible.

2.1.9.14 PMOPT

Define the options for the performance monitor.

2.1.9.15 PREPSUBS

Define and prepare BS2000/OSD dynamic subsystems for CA-IDMS system or user application modules. Refer to 5.4, "Program Loading" on page 5-11 for more details.

2.1.9.16 REMOVAPA

This is the inverse function of the APPLYAPA function: an applied APAR is removed.

2.1.9.17 RHDCCTAB

Secure DCMT commands. Refer to the *CA-IDMS Security Administration* manual for details.

2.1.9.18 RHDCOPTF

Apply Optional APARs. Refer to the *CA-IDMS System Operations* manual for details.

2.1.9.19 RHDCSRTT

Define and generate the security load module. Refer to the *CA-IDMS Security Administration* manual for more details.

2.1.9.20 RHDCUXIT

Define and generate the system exits. Refer to the *CA-IDMS System Operations* manual for more details.

2.1.9.21 SETDEFLT

Set default values for the parameters of most of the JCL procedures.

2.1.9.22 STGIDCMD

Define the available BS2000/OSD commands in the STGID DC task.

2.1.9.23 SVCOPT

Edit the ERE extension size for special collection of CA-IDMS resource usage statistics. Refer to the *CA-IDMS System Operations* manual for more details.

2.1.9.24 USRIDXIT

Define and generate the user ID exit to extract the userid in batch or dialog programs. Refer to the *CA-IDMS System Operations* manual for more details.

2.1.9.25 WTOEXIT

Edit, modify, compile and link the WTOEXIT. Refer to the *CA-IDMS System Operations* manual for more details.

2.1.9.26 WTOREXIT

Edit, modify, compile and link the WTOREXIT. Refer to the *CA-IDMS System Operations* manual for more details.

2.1.10 OLQBATCH

This procedure contains the JCL to execute CA-OLQ in batch, central or local mode. The procedure can be called by jobs entered thru CA-OLQ on-line.

To enter jobs in batch from CA-OLQ, these steps are required:

1. Modify SYSGEN with an OLQ statement, which specifies a BATCH CLASS different from 0 (maximum is 64).
2. Define an OLQBATCH-JCL QFILE in the IDD containing the JCL to run OLQ in batch. For example:

```

ADD QFILE NAME OLQBATCH-JCL LANGUAGE IS OLQ
  MODULE SOURCE FOLLOWS
  /.OLQBATCH LOGON TIME=300,PRIORITY=(,EXP)
  /CALL-PROC NAME=*LIB-ELEM(LIB=dba.source.lib,ELEM=FILE),      -
  /  PROC-PAR=(MODE=CENTRAL)                                   -
  /CALL-PROC NAME=*LIB-ELEM(LIB=dba.source.lib,ELEM=OLQBATCH), -
  /  PROC-PAR=(PREF=prefix,                                   -
  /             LSTF=T.SYSLST,OUTF=T.SYSOUT,SortLIB=$SortLIB, -
  /             NOTE=Y,                                       -
  /             USER=JONES,                                   -
  /             MSG="'HI JONES, YOUR JOB HAS FINISHED |'")
  <<OLQ>>
  /LOGOFF
  MSEND.

```

The example of the JCL prototype explains most of the parameters of the procedure. Edit the procedure for more details.

3. Install the Spool-Interface as described in Chapter 7, “Generalized Spool Interface.”
4. Add the print class of the OLQ statement to the LTERM used by the Spool-Interface.

Refer to the *CA-OLQ Reference Guide* for more details.

2.1.11 PLUS

This procedure creates a load module containing information on how to load the programmable keys on 975X terminals using the BS2000/OSD utility PLUS. The input for the default module (BS2KP975) created during the installation, is contained in the procedure.

2.1.12 PRLG

This procedure offloads the log.

2.2 Notes for the Database Administrator

2.2.1 AJNL and PRLG ENTER Files

During installation, two ENTER files are created for the system: one calls the "AJNL" DBA source library procedure, the other calls the "PRLG" DBA source library procedure. Two ADD-FILE-LINK commands are issued by the FILE procedure, with MODE = CV:

```
/ADD-FILE-LINK LINK-NAME=AJNL,F-NAME=ajnl-filename  
/ADD-FILE-LINK LINK-NAME=PRLG,F-NAME=prlg-filename
```

The WTOEXIT routine automatically enters the file with linkname AJNL or PRLG when a journal, respectively a log file has to be offloaded.

Note: If WTOEXIT does not find a TFT entry with linkname AJNL or PRLG, it tries to enter the file named E.AJNL, respectively E.PRLG.

2.2.2 GSPI Enter File

An ENTER file to start the generalized spool interface is created:

prefix.E.GSPI

For more information, refer to Chapter 7, "Generalized Spool Interface."

2.2.3 APARs

In the CA-IDMS APAR library, these members contain useful information:

Member Name	Description
GJF0APAR	Short description of the APARs on the tape
GJF0XREF	Relationship between official APARs and their corresponding USERMODs
GJF0OPTN	Description of the optional APARs
OPTTABLE	A cross-reference between optional APARs and it's corresponding RHDCOPTF bit
BS2KREPS	Actual APARs

APAR Format: The APAR must be a source member in the APAR library and use this format:

```

++APAR      (aparnum)
++VER       (Z038)      FMID(compont)
++ZAP       (objname)  DISTLIB(DISTLOAD)
NAME        csectnam
IDRDATA     aparnum
BASE        000000
VER AAAAA  VVVV,VVVV
REP AAAAA  RRRR,RRRR

```

Parameter	Description
<u>aparnum</u>	Number of the APAR
<u>compont</u>	Component to which the object "objname" belongs
<u>objname</u>	Name of the object to REP
<u>csectnam</u>	Name of the CSECT to REP
AAAAA	Address relative to the start of the CSECT
VVVV	Verify information
RRRR	Repair information

Verify and repair information can be up to 16 bytes and must match in length. All LMS functions can be used to manage APARs.

Applying Optional APARs: Search GJF0OPTN for the desired functionality and note the APAR number. Next, Search the OPTTABLE for the APAR number. If found:

1. Note the corresponding RHDCOPTF bit number.
2. Use IDMSMOD function RHDCOPTF to edit, assemble and link a new RHDCOPTF module containing the optional functionality.

If the APAR number is not found:

1. Extract the optional APAR from BS2KREPS and save it in your APAR library using the APAR number as the member name.
2. Use IDMSMOD function APPLYAPA to apply the optional APAR.

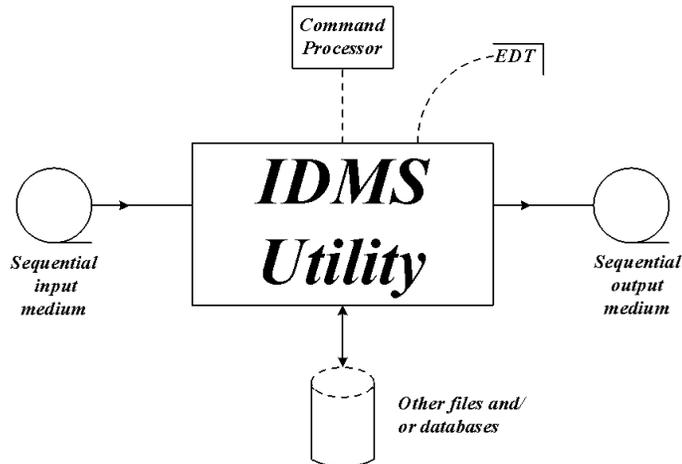
For more information about:

- Procedures IDMSMOD, APPLYAPA and RHDCOPTF, see 2.1, "JCL Procedures" on page 2-3
- RHDCOPTF, see *CA-IDMS System Operations*

2.3 IDMS Utility Interface with BS2000/OSD

The CA-IDMS utilities referred to in this section include the compilers, such as IDMSCHEM, IDMSUBSC, IDMSDDDL, RHDCSGEN, RHDCMAP1, RHDCMPUT, IDMSBCF, the "Batch Command Facility" and the remaining utilities such as IDMSLOOK and IDMSRPTS.

The section discusses the access and use of the BS2000/OSD system files SYSDTA, SYSLST, SYSOUT and SYSOPT, the calling of the BS2000/OSD interactive editor EDT, and the BS2000/OSD Command Processor. A simplified model of the CA-IDMS utilities and their interface with BS2000/OSD is illustrated as follows:



The sequential input medium usually contains the input control statements, such as SQL statements for IDMSBCF or DDDL statements for IDMSDDDL, while the output medium is used to report on the execution of the utility. The default input medium is the BS2000/OSD system file SYSDTA, the default output medium is the BS2000/OSD system file SYSLST.

The standard properties of SYSDTA are also applicable when IDMS utilities are reading from SYSDTA. For example, SYSDTA can be assigned to a terminal and reassigned with BREAK, ASSIGN-SYSDTA TO=filename.

Instead of using the appropriate ASSIGN-SYSxxx command to assign SYSDTA, SYSOPT or SYSLST, it is possible to use the ADD-FILE-LINK command with the following link names: SYSIPT (for SYSDTA), SYSOPT or SYSLST. These link names **have precedence** over the respective ASSIGN-SYSxxx assignments. In this example, the input is read from input2:

System File Assignment Example

```
/ASSIGN-SYSDTA TO=input1
/ADD-FILE-LINK LINK-NAME=SYSIPT,FILE-NAME=input2
```

To enhance reassignment of both input and output media, such as using EDT as a combined interactive input-output medium, a number of control statements can be inserted in the input. An input record is recognized as a control statement if one of these keywords appear in column 1 of the input record:

=COPY	Reassign input to library member
=EDT	Call EDT as a subroutine
=END	Close ON-EDT-RETURN or ON-EDT-EOF file
=IPT	Reassign input
=ON-EDT-EOF	Open file ON-EDT-EOF for output
=ON-EDT-RETURN	Open file ON-EDT-RETURN for output
=LST	Reassign output
=SYS	Pass a BS2000/OSD command to the ISP command processor

1. =COPY

It is possible to reassign input to an element of an LMS library, using the "=COPY IDMS" function. This function reassigns implicitly the SYSDTA system file to the library member specified on the card.

Note: When the =COPY statement is read, SYSDTA must be assigned to *PRIMARY or *SYSCMD.

=COPY JCL Example

```
/ADD-FILE-LINK LINK-NAME=linkname1,FILE-NAME=library1
/ADD-FILE-LINK LINK-NAME=linkname2,FILE-NAME=library2
```

=COPY INPUT Example

```
=COPY IDMS linkname1.member-name1
=COPY IDMS linkname2.member-name2
```

2. =EDT

With this control statement, the utility passes the EDT-statement(s), following the keyword =EDT to the BS2000/OSD EDT editor.

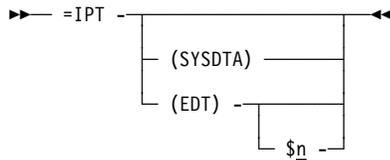
Program control is in EDT until all EDT-statements are processed or, in case the DIALOG statement has been issued, until EDT reads a (@)RETURN, (@)HALT or K1 function key.

When control returns to the CA-IDMS-utility, the utility reassigns the input medium to the file with linkname OEDTRET. The file is read until end-of-file or until new reassignment of the input medium and the input is processed.

These statements are needed if applications are built, with complex use of EDT, within a CA-IDMS utility.

3. =ON-EDT-EOF - This control statement opens the file with linkname OEDTEOF for output. The input is copied to the file OEDTEOF until the control statement =END is read.
4. =ON-EDT-RETURN - This control statement opens the file with linkname OEDTRET for output. The input is copied to the file OEDTRET until the control statement =END is read.
5. =END - Closes the OEDTEOF or OEDTRET file.
6. =IPT

This function reassigns the input for the utility. The syntax is the following:



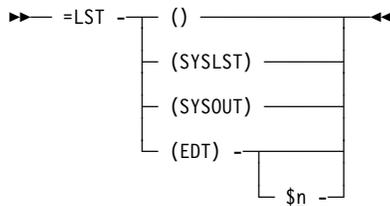
Parameter	Purpose
n	The EDT virtual file number. (Default for n is 0.)

=IPT (EDT) \$7

The input is reassigned to the EDT virtual file number 7. When EOF is reached on the EDT virtual file, the input medium is automatically reassigned to the file with linkname OEDTEOF and the input is processed.

7. =LST

This function (re)sets the output of the utility. The syntax is the following:



Parameter	Purpose
n	The EDT virtual file number. (Default for n is 1.)

8. =SYS

This invokes the CMD command processor. =SYS does an BKPT SVC

=SYS command executes the BS2000/OSD command command

=SYS Example

=SYS ADD-FILE-LINK LINK=INLIB,F-NAME=libname

Note: The IDMSMOD function EDTENVIR is automatically executed during installation and creates a default OEDTRET and OEDTEOF file. The EDT environment is configured in this manner:

After the utility is initialized, an interactive session with \$EDT is started. The input is expected in virtual file \$0, control is returned to CA-IDMS's utility with Halt, Return or K1. The output is in virtual file \$1, while virtual files \$5 and \$6 are cumulative input and output. By changing the contents of the OEDTRET and OEDTEOF files, the DBA or the user can define their own EDT environment.

2.4 BS2KOBJM Utility

The BS2KOBJM utility can be used to:

- Convert object modules from CA-IDMS load area format to LMS type R format.
- Force the RMODE and/or AMODE attribute to a specific value.

Input parameters are read from the SYSDTA system file. Its syntax is the following:

MODULE=module-name [,VERSION=version] [,RMODE=ANY/24] [,AMODE=ANY/31/24]

Input Module: An input object module can be extracted from different mediums:

- First, the utility tries to read object module from a variable SAM file assigned to the OBJMINPT linkname.
- If the file can't be opened, the utility extracts the object member from the LMS object library assigned to the OBJMLIBI linkname using the name specified in the MODULE= parameter.

When the input module is extracted from the OBJMLIBI library, more than one module can be processed by the utility in the same run.

Output Module: An output module is saved in an LMS object library assigned to the OBJMLIBO linkname. The member name is the same as the module-name specified in the MODULE= parameter.

Run Termination: When the utility terminates, an explicit release of OBJMINPT, OBJMLIBI and OBJMLIBO linknames is done.

2.5 Tapes and Sequential Files

Utilities processing tapes also run similarly with disk files.

The rollforward utility (IDMSBCF statement ROLLFORWARD) uses the BTAM access method for tapes. BTAM can not handle multi-volume files; when encountering an end of volume condition, this message is displayed on the console:

```
IDMS000 Is this the last volume to be processed ? Reply (Y=YES;N=NO)
```

The operator must reply to the message with 'Y' or 'N'.

- 'N' - the next volume is requested
- 'Y' - the utility terminates

When encountering a run-unit that must be rolled back, the tape is read backward. If the abort run-unit extends over more than 1 volume, this message is displayed on the console:

```
IDMS000 Mount PREVIOUS volume. Reply when volume mounted (ETX=YES;N=NO)
```


Chapter 3. Address Space Layout

- 3.1 Introduction 3-3
- 3.2 Startup Processing 3-4
 - 3.2.1 Startup Phase 1 3-4
 - 3.2.2 Startup Phase 2 3-5
- 3.3 Startup Parameters 3-7
 - 3.3.1 Example 3-8

3.1 Introduction

This section describes step-by-step how the CA-IDMS region is built and the allocation of memory during startup. There is also a discussion of system messages that are displayed on the console and their role in assisting you with tuning the CA-IDMS system.

For information about the steps required for starting up a central version or implementing multi-tasking, please see the appropriate sections in the *CA-IDMS System Operations* manual.

3.2 Startup Processing

This section describes how the CA-IDMS region is built during the startup processing. It also considers the best values the user can assign to the REGION and FREESTG startup parameters to build the most compact address space for a central version.

The following figure gives an overview of the address space layout as used by a central version.

ADDRESS	COMMENTS	Ref. NR.
0	BS2KSTAR during startup; TSN-specific table	1
1000	IDMSLBS2; IDMSDCB IDMS non XA-system: <ul style="list-style-type: none"> ■ tables ■ program pools ■ storage pools ■ ... 	2 3
DC-HIADDR	not used RHDCNTRY FREESTG Database buffers during warmstart	4 5 6 7
REGION	Free class 5 & 6 memory Non-XA part of the operating system	8
16M	CCI memory pool	
	ESE/ERE memory pool	
	IDMS XA system: <ul style="list-style-type: none"> ■ tables ■ XA program pools ■ XA storage pools ■ ... 	9

3.2.1 Startup Phase 1

Running the BS2KSTAR program starts central version. This program is contained entirely in the first segment memory (64 Kbytes); this area is TSN dependent when running the CV in a multi-tasking environment (see area #1 in the address space layout schema). These steps are executed:

1. Read the BS2KSTAR specific parameters. The REGION and FREESTG parameters must be provided if the user wants to override the values generated in the #DCPARM macro.
2. BS2KSTAR enables the global memory pool containing the system tables, the non-XA nucleus modules, and the CA-IDMS non-XA program and storage pools. This memory pool starts at the second segment memory address and its length is

equal to the REGION parameter value truncated to a multiple of a segment, minus the size of the first segment memory, i.e. areas #2 + #3 + #4 + #5 + #6.

3. BS2KSTAR loads the startup module, by default IDMSDCB, at the beginning of the global memory pool.
4. BS2KSTAR enables the CCI memory pool. Refer to the appropriate chapter in the CA-IDMS Server guide for more information.
5. BS2KSTAR gives control to the startup module.

3.2.2 Startup Phase 2

From this point, the space occupied by the BS2KSTAR load module is reused by system tables and by information that requires to belong to a TSN-dependent area when running in a multi-tasking environment. The startup continues with these steps:

1. All the space contained between the end of IDMSDCB load module and the end of the global memory pool, minus the values specified for the FREESTG parameter is acquired, i.e. areas #3 + #4 + #5.
2. The RHDCNTRY load module is loaded at the end of this area (area #5); it takes control to process the next actions of the startup.
3. The WARMSTART processing uses local run-units to recover the database; "local" means the database buffers are not requested within the global memory pool, but after it. The WARMSTART processing does not need as many buffers as declared in the DMCL. It requires a maximum of 3 buffers for each database buffer pool. Database buffers are always requested one by one, i.e. a multiple of 4K is always allocated to each buffer.
4. The XA pools, if requested, are initialized (area #9): a unique memory pool is enabled above the 16MByte line for all the XA pools. The size of this memory pool is a multiple of 1MByte, but only the space needed for the pools is allocated.
5. The system tables, the program pools and the storage pools are built in the global memory pool, starting at the end of module IDMSDCB (see area #3).
6. The different lines are opened. Storage in the reserved FREESTG area (area #6) is acquired by some open driver modules for this processing: the batch simulator is an example.
7. The system releases the global memory pool space that is not used, including the space occupied by the RHDCNTRY module. The areas #4 + #5 + #6 are now making a new free area in the global memory pool. Space within this area is allocated for the name manager, the database and journal buffers, and for other needs, like for example, a trace table when running multi-tasking.
8. In a multi-tasking environment, different ENTER-jobs are started to satisfy the requested number of TSN's (MAXTSN parameter from BS2KSTAR). These new jobs use the first memory segment (area #1) for their own requirements, such as the TSN dependent information. They enable all memory pools that are already enabled by the MAINTASK.

9. Finally, the ERE memory pool is built (area #7) in XA storage. In a multi-tasking environment, the ERE memory pool is enabled by all the subtasks.

3.3 Startup Parameters

The FREESTG area (area #6) is not intensively used by the startup processing. For all configurations, 64K should be a satisfactory value. With this in mind, only the REGION parameter can still influence the address space layout; later, if changes are made to the size of the CA-IDMS pools or to the number of database buffers, only this parameter must be modified to allow a new startup.

Information Returned by the Startup: During the startup processing, some DC messages are displayed into SYSOUT, with useful information about the space needed by the system. These are the messages:

DC390005 RHDCPARM FREESTG RELEASED

Explanation: This is the value of the FREESTG parameter declared in the #DCPARM macro or the override value specified with the FREESTG input card of BS2KSTAR. This value should always be 64.

DC390006 REGION NEEDED TO STARTUP

Explanation: This is the value obtained by adding the size of the areas #1 + #2 + #3 + #5 + #6 + #7 from the layout schema and by rounding it up to a multiple of 64K.

DC390007 SYSTEM CONFIGURATION SIZE

Explanation: This value is equal to the "Region needed to startup" - "Rhdcparm freestg released", i.e. the difference between the values displayed by the two previous messages.

DC390008 STORAGE RETURNED TO OPSYS

Explanation: This is the space within the global memory pool that is not used by the startup, plus the space occupied by the RHDCNTRY module (areas #4 + #5).

DC390020 TOTAL AMOUNT OF XA STORAGE

Explanation: This message is only displayed on a system where XA pools are defined.

Defining the REGION Parameter: The second DC message "REGION NEEDED TO STARTUP", should be the most appropriated value to assign to the REGION parameter to start a system with the smallest possible size. But due to the fact that:

- The FREESTG value is fixed to 64K, and therefore, doesn't reflect the total size that is used for the database buffers
- Those buffers, together with all other areas, are not yet requested by the system
- The space used by the RHDCNTRY module can also be used for the buffers

A more accurate value for the REGION parameter can be evaluated from the information returned by the execution of the STGID task code, with the DISPLAY ADDRESS SPACE option:

STGID D ADS

The correct REGION size can be obtained by adding the following values returned by STGID; this sum is rounded up to the next segment boundary.

```
REGION value =  DC-SYSTEM HIADDR value
                + NAME MANAGER AREA size
                + DATABASE BUFFERS size
                + JOURNAL BUFFERS size
                + OTHERS size
                + FREESTG INPUT VALUE (i.e. 64K)
```

The space occupied by the RHDCNTRY module is not considered in this formula, because most of these areas are requested after the module is deleted; therefore, they are also using the space it occupied.

3.3.1 Example

This example displays the output returned by a STGID D ADS command executed on a system started with REGION=8000 and FREESTG=64.

REGION INPUT VALUE	8000 K	DC-SYSTEM HIADDR	-	5204 K
FREESTG INPUT VALUE	64 K	NAME MANAGER AREA	N	8 K
ACQUIRED AT STARTUP	7808 K	DATABASE BUFFERS	B	0 K
FREED AFTER STARTUP	2604 K	JOURNAL BUFFERS	J	0 K
MAX SPACE BELOW 16MB	14336 K	FREE IN MEMORY POOL	+	2660 K
USER SPACE BELOW 16MB	13312 K	ERE MEMORY POOL	E	128 K
MAX SPACE ABOVE 16MB	1978 M	OTHERS	#	16 K
XA STORAGE AREA SIZE	7980 K	UNUSED SPACE	.	7960 K
XA STORAGE AREA ADDR	02500000			

1 CHARACTER REPRESENTS 1 MEMORY PAGE OF 4 K : SUBTASK=MAINTASK

0000	-----	256 K
40000	-----	512 K
80000	-----	768 K
C0000	-----	1024 K
100000	-----	1280 K
140000	-----	1536 K
180000	-----	1792 K
1C0000	-----	2048 K
200000	-----	2304 K
240000	-----	2560 K
280000	-----	2816 K
2C0000	-----	3072 K
300000	-----	3328 K
340000	-----	3584 K
380000	-----	3840 K
3C0000	-----	4096 K
400000	-----	4352 K
440000	-----	4608 K
480000	-----	4864 K
4C0000	-----	5120 K
500000	-----+++++	5376 K
540000	+++++	5632 K
580000	+++++	5888 K
5C0000	+++++	6144 K
600000	+++++	6400 K
640000	+++++	6656 K
680000	+++++	6912 K
6C0000	+++++	7168 K
700000	+++++	7424 K
740000	+++++	7680 K
780000	+++++NN+++++.UUUUUUUUUUUU	7936 K
7C0000	UUUUUUUUUUUUUU###.....	8192 K
800000	8448 K
840000	8704 K
880000	8960 K
8C0000	9216 K
900000	9472 K
940000	9728 K
980000	9984 K
9C0000	10240 K
A00000	10496 K
A40000	10752 K

3.3 Startup Parameters

A80000	11008 K
AC0000	11264 K
B00000	11520 K
B40000	11776 K
B80000	12032 K
BC0000	12288 K
C00000	12544 K
C40000	12800 K
C80000	13056 K
CC0000	13312 K

Chapter 4. CA-IDMS/DB

- 4.1 Alignment Requirements 4-3
- 4.2 COBOL Database Procedures 4-4
- 4.3 SYSIDMS Support 4-5
- 4.4 Interface for Application Programs 4-6
 - 4.4.1 Primary Interface Module 4-6
 - 4.4.2 Streamline Batch to CV Interface 4-7
- 4.5 QSAM Facility 4-9

4.1 Alignment Requirements

- The record SUBSCHEMA-CTRL / SSCTRL must be aligned on a doubleword BOUNDARY. For Assembler use a "DS 0D". In COBOL, this is achieved by a 01 level definition; if SUBSCHEMA-CTRL is not an 01 level, it should be made doubleword aligned.
- Fields into which a database key or any other binary value is accepted, must be aligned on a full word boundary. These fields must be defined as "PIC S9(8) COMP SYNC" in COBOL.

4.2 COBOL Database Procedures

COBOL database procedures that are not linked with the subschema, must be declared as Assembler programs. The link-edit parameters are dependent on how the program is linked.

- Linked with BINDER

```
//INCLUDE-MODULES LIBRARY=idms.loadlib,ELEMENT=BS2KENTR,TYPE=(L,R)  
//RENAME-SYMBOL BS2KV$C$,cobol-program-name
```

- Linked with TSOSLNK

```
RENAME BS2KV$C$,cobol-program-name  
INCLUDE BS2KENTR,idms.loadlib
```

4.3 SYSIDMS Support

SYSIDMS parameters can be specified in these ways:

1. Using

```
/ADD-FILE-LINK LINK-NAME=SYSIDMS,FILE-NAME=filename
```

2. Using

```
/ADD-FILE-LINK LINK-NAME=SYSIDMS,FILE-NAME=*DUMMY
```

The SYSIDMS parameters are read from SYSDTA until a record with END-SYSIDMS is read.

This method is not advisable for application programs also reading data from SYSDTA. Indeed, it can be confusing to know when CA-IDMS is reading SYSDTA or when the application program is reading SYSDTA.

3. Define a job variable with linkname *SYSIDMS.

Note: When using job variables, the file with linkname SYSIDMS is not read.

For more information about the SYSIDMS parameters, refer to *CA-IDMS Database Administration* manual.

4.4 Interface for Application Programs

4.4.1 Primary Interface Module

On BS2000/OSD, an IDMS primary interface module exists for the batch, DC-batch and DC-COBOL programs. Its goal is to minimize the amount of system coding linked with the user applications, and to offer a definitive interface, independent of CA-IDMS's release. Programs linked with this interface are upwardly compatible with CA-IDMS.

How it Works: The primary interface module, called IDMSPBS2, is linked with the user application. At runtime,:

1. The new secondary interface module, IDMSSBS2, is loaded via a BIND SVC at the first CA-IDMS call. IDMSSBS2 is loaded from the load library associated with the CDMSLODR linkname, or from the BLSLIBnn chain if it is not found in the first library.
2. Next, the CA-IDMS request is passed to the real interface module:
 - IDMS - Batch
 - IDMSDCBI - DC-BATCH
 - IDMSCOBI - DC-COBOL

The secondary interface module, IDMSSBS2, is composed of these object modules:

- IDMS
- IDMSCANC

and optionally,

- IDMSOPTI

IDMSOPTI Module: A user IDMSOPTI module may be linked with the user application and/or with the secondary interface module; the IDMSOPTI linked with the user application always takes precedence.

IDMSCANC Module: The standard IDMSCANC module is linked with the secondary interface. As for IDMSOPTI, another IDMSCANC may be linked with the user program. To take precedence on the other, this module must be included before the IDMSPBS2 object in the user application load module.

COMP Procedure: The COMP procedure delivered in the source library includes the new interface module when linking a user application program.

4.4.2 Streamline Batch to CV Interface

Streamline Batch to CV Interface: A separate IDMSBS2 is in the PLAM library "INTBLOAD.LOADLIB" for batch programs executing against a central version running CA-IDMS Release 14.0 or later.

This high performance interface may be used with programs utilizing only CA-IDMS 10.21 features and does not support the use of the new CA-IDMS 12.0 features such as SQL access. This interface is not applicable to local batch jobs. This separate secondary interface IDMSBS2 can be activated by assigning the PLAM library INTBLOAD.LOADLIB to the CDMSLODR linkname or from the BLSLIBnn chain if it was not found in the first library. Because of this, this IDMSBS2 must be kept separate from your regular CA-IDMS load modules. When you release SYSCTL, the old IDMSBS2 module loads a standard local mode environment and the batch programs run in local mode.

This separate secondary interface module IDMSBS2 is composed of these objects:

- IDMS (from S10214)
- IDMSCANC

and optionally,

- IDMSOPTI.

IDMSOPTI Module: A user IDMSOPTI module may be linked with the user application and/or with the secondary interface module; the IDMSOPTI linked with the user application takes always precedence.

IDMSCANC Module: The standard IDMSCANC module is linked with the secondary interface. As for IDMSOPTI, another IDMSCANC may be linked with the user program. To take precedence on the other, this module must be included before the IDMSPBS2 object in the user application load module.

IDMSMOD Function BTCIDXIT: This user ID exit is called when the 10.2 streamlined batch interface is used in a CA-IDMS 14.0 and above environment. It makes it possible to supply a user ID with access a CA-IDMS 14.0 and above central version from a batch program.

Two members are added to your DBA source library:

- **BTCIDXIT, type S**

This example shows how to provide the user ID. When the exit is called, the address of the ERE-extension is passed in R1. The exit must supply the user ID in the EREXUSER field, an 18 byte field, based upon the #UCFDS DSECT.

- **BTCIDXIT, type J**

This procedure compiles, and link-edits the module and the IDMSINTB load module found in the CA-IDMS load library. The BTCIDXIT module and the

relinked IDMSINTB load module are saved in the DBA load library. This procedure must be called via the IDMSMOD function BTCIDXIT.

4.5 QSAM Facility

The QSAM facility provides a mechanism to optimize sequential access (area sweep) of data in local mode. These SYSIDMS parameters are involved when using QSAM:

- IDMSQSAM=ON - Activates the feature
- QSAM#BUF=nnn - Allows optional fine tuning
- QSAMAREA=name - Allows optional specification of the area

For more information on these SYSIDMS parameters, refer to *CA-IDMS Database Administration*.

Note: All files subject to QSAM processing require SHARED-UPDATE=*YES in their ADD-FILE-LINK JCL command. Failing to specify SHARED-UPDATE=*YES can result in DMS error 0D99 or a 2230 abend.

Chapter 5. CA-IDMS/DC

5.1 Program Function Keys (PF-Keys)	5-3
5.2 Special Considerations Regarding TRANSDATA Devices	5-7
5.3 UCF Considerations	5-9
5.3.1 Generalities	5-9
5.3.2 UCF-BATCH	5-9
5.3.3 UCF-TIAM	5-10
5.3.4 UCF-UTM	5-10
5.4 Program Loading	5-11
5.4.1 Library Search Chain	5-11
5.4.2 Notes for Central Version	5-11
5.4.3 Dynamic Subsystem Support	5-12
5.4.3.1 What is it?	5-12
5.4.3.2 Why Use Subsystems?	5-12
5.4.3.3 When Not to Use Subsystems?	5-12
5.4.3.4 Installing CA-IDMS Subsystems	5-13
5.4.3.5 Which Modules Can be Placed in a Subsystem?	5-14
5.5 Recovery from Program Errors	5-15
5.5.1 Trapping of Termination SVC	5-15
5.5.2 Recovery from Alignment Errors	5-15
5.6 Storage Protection	5-16
5.6.1 Description	5-16
5.6.2 Performance Considerations	5-16
5.6.3 FETCH Protection	5-16
5.6.4 Installation on BS2000/OSD OSD V2.0	5-16
5.6.5 Installation on BS2000/OSD OSD V3.0	5-16
5.7 STGID Task	5-18
5.8 XS Support	5-20

5.1 Program Function Keys (PF-Keys)

For simplicity and ease of understanding, all CA-IDMS documentation materials normally refers to only 2 types of terminals:

- 3270-type - These are the 327x and the 317x. These terminals have enhanced hardware functions, such as protected fields, different visual attributes, program function keys, etc.
- Others - mainly TTY. The I/O to these terminals is line oriented.

The Siemens terminal 975X (9748, 9750, 9751, 9752, 9755, 9756, etc.) has the same hardware functions as the 3270-type. These terminals also support two "extended attributes", blinking and underlining (or reversed and italic). All other Siemens terminals are classified as non-3270 terminals and they are supported as line terminals.

Note: Extended attributes are only available on a 3279-type IBM terminal.

To make full use of the PROGRAM FUNCTION KEY (PFK) support in CA-IDMS DC/UCF the programmable keys of the 9750 terminal must be loaded by the "DCKEYS" task.

Terminology: There is a difference in terminology between IBM's 327x environment and Siemens 975x terminals. In the 327x environment there are two types of keys:

- Attention keys — ENTER, PA1, PA2, PA3 and CLEAR
- Program function keys — PF1 - PF24

In the Siemens terminal environment there are three types of keys:

- K-keys
- F-keys
- Programmable keys (P-keys)

This table shows the relationship between the 3270 transmission keys and the equivalent transmission and programmable keys of the Siemens 975X terminals.

PF-KEYS

3270 Key	975X Key ¹	975X Key	975X P-key ²
ENTER	DU	DU	N/A
CLEAR	K3	K3	N/A
PA1	K1	K1	N/A
PA2	K2	K2	N/A
PF1	F1	F1	P1

5.1 Program Function Keys (PF-Keys)

3270 Key	975X Key 1	975X Key	975X P-key 2
PF2	F2	F2	P2
PF3	F3	F3	P3
PF4	F4	F4	P4
PF5	F5	F5	P5
PF6	K4	F6	P6
PF7	K5	F7	P7
PF8	K6	F8	P8
PF9	K7	F9	P9
PF10	K8	F10	P10
PF11	K9	F11	P11
PF12	K10	F12	P12
PF13	K11	F13	P13
PF14	K12	F14	P14
PF15	K13	F15	P15
PF16	K14	F16	P16
PF17	K15	F17	P17
PF18	K16	F18	P18
PF19	K17	F19	P19
PF20	K18	F20	P20
PF21	K19	F21	N/A
PF22	K20	F22	N/A
PF23	K21	F23	N/A
PF24	K22	F24	N/A

Note:

1 - For historical purposes 2 mappings exist. This mapping is used only on devices that do not support F6-F24.

2 - Older devices might have less than 20 P-keys.

DCKEYS: DCKEYS is the name of the task loading the programmable keys on 975X terminals. DCKEYS loads the programmable keys of a 975X terminal with the values as specified in a "PLUS"-module.

The syntax for the execution of the DCKEYS task is as follows:

```
DCKEYS    [ 9750 <--          ] [ ,DOC ]
           [ plus-module-name ]
```

If no value is specified for the first positional parameter, 9750 is assumed. In this case DCKEYS, loads the P-keys with the values found in the PLUS-module "BS2KP975", specifying 9750 has the same effect.

The previous table lists the assignments of the module BS2KP975 after the standard installation. Site specific copies of BS2KP975 can be made with the procedure PLUS, contained in the source library. The PLUS-modules, created by the procedure PLUS, are loaded by the system from the loadlibs in the runtask program pool when DCKEYS is executed.

The source for the generation of the "PLUS" load module BS2KP975 is contained in the PLUS procedure. The sources of the Release 10.xx PFkey support, are also in the PLUS procedure and are used if the PLUS module BS2KO975 must be generated.

User specific PLUS-modules are also possible.

The second positional parameter, DOC, triggers DCKEYS to document how the P-keys are loaded.

Note: DCKEYS are only documented when DOC is specified.

To acknowledge the loading of the P-keys the MSG-function can be used when creating the PLUS-module.

Please refer to the Siemens manuals on the PLUS utility for further information.

BS2KDKEY: The CA-IDMS/DC task DCKEYS invokes the program BS2KDKEY. This program can be called from any CA-IDMS/DC program, including an ADS dialog. The program resides in the CA-IDMS load library and must be called using the DC or ADS conventions (LINK, TRANSFER CONTROL, etc.).

One parameter must be specified with following structure in COBOL format:

01	DCKEYS-COM-AREA.		
03	DCKEYS-PLUS-MODULE	PIC X(8)	
		VALUE ' '.	
03	DCKEYS-STATUS	PIC X(4)	
88	DCKEYS-OK	VALUE '0000'.	
88	DCKEYS-NO-PKEYS	VALUE '0100'.	
88	DCKEYS-NO-LTE	VALUE '0200'.	
88	DCKEYS-NO-PTE	VALUE '0300'.	
88	DCKEYS-GETSTG-ERROR	VALUE '10xx'.	
88	DCKEYS-LOAD-ERROR	VALUE '11xx'.	
88	DCKEYS-TREQ-ERROR	VALUE '12xx'.	
03	DCKEYS-OPTIONS	PIC X .	
88	DCKEYS-OPT-DEFAULT	VALUE ' '.	
88	DCKEYS-OPT-DOC	VALUE 'D'.	
03	DCKEYS-FILLER	PIC X(3) VALUE ' '.	

Explanation of Fields:

DCKEYS-PLUS-MODULE: Input parameter. Must contain the name of the PLUS module to be used. If this field is left blank, the program uses the BS2KP975 PLUS module.

DCKEYS-STATUS: Output parameter. If successful, the status is set to DCKEYS-OK. Meaning of the different error status:

- NO-PKEYS - The terminal device has no programmable P-keys.
- NO-LTE - The task running the BS2KDKEY program has no logical terminal associated to it.
- NO-PTE - The task running the BS2KDKEY program has no physical terminal associated to it.
- GETSTG-ERROR - DC returned an error on the GET STORAGE function; xx specifies the error.
- LOAD-ERROR - DC returned an error on the LOAD function; xx specifies the error. This occurs when the PLUS module does not exist.
- TREQ-ERROR - DC returned an error on the WRITE TO TERMINAL function; xx specifies the error.

DCKEYS-OPTIONS

- OPT-DEFAULT - Default option
- OPT-DOC - The content of the P-keys is displayed on the terminal.

5.2 Special Considerations Regarding TRANSDATA Devices

- Mapping supported terminals (975X/ADDS) are processed like IBM 3270 terminals. This means each field starts with a protected space position to simulate the 3270 attribute byte.

Suggested values for terminal hardware 975X: ASZ (visual attribute) should be fixed coupled with FBZ (protect attribute). This is done by setting to 1 bit 4 of switch S5 for 9750's. On 9750's, underlined fields seems to be 1 position to long if the coupling is not set. On 9755 and 9756 the effect can be worse. For 9755 and 9756 there is an alternative, if this coupling is not wanted. Define the PRESOPT=IS2AP3 in the logmode entry for the sessions (connections) with 9755 and 9756. This option forces mapping to use pairs of IS2 and IS3. Refer to 6.2.2, "The Logmode Table" on page 6-7 for more information about the PRESOPT parameter. This approach also corrects some errors in the 9755 firmware.

For 9750 and 9752 terminals there is a restriction on the number of fields that can be defined per line. There is no restriction for 9755 or 9756 terminals.

The number of map fields that can be defined per line of the screen is given by the following formula:

$$12 \leq \text{maximum total number of fields} \leq 24$$

The following equation must be fulfilled:

$$(\#FD.UNP.DEL * 4) + (\#FD.UNP * 3) + (\#FD.PROT * 2) = 48$$

where

#FD.UNP.DEL	=	# UNPROTECTED FIELDS, W DELIM.
		+ # PROTECTED FIELDS, MDT SET, W DELIM.
#FD.UNP	=	# UNPROTECTED FIELDS W/O DELIM.
		+ # PROTECTED FIELDS, MDT SET, W/O DELIM.
		+ # PROTECTED FIELDS, MDT NOT SET, W DELIM.
#FD.PROT	=	# PROTECTED FIELDS, MDT NOT SET, W/O DELIM.

- Large maps can require CA-IDMS/DC to split messages. Normally, CA-IDMS DC/UCF splits the datastreams depending on the maximum length returned by DCAM/TIAM. However, there are configurations where this maximum still is too big. Especially with terminals connected thru a MSN the maximum doesn't seem to be correct. The MAXLN parameter in the logmode can be used in this case, to reduce the maximum to a correct value.

Another approach is turning on the COMPACT option in the SYSGEN of the line. This option is advisable in any case, because it reduces the size of the datastreams.

- Do not use the LZE key for DC-MAPS.

Basic Mode in DC: There are two kinds of basic mode:

- Device independent. This occurs when a DC-program builds basic 3270-datastreams. The application program remains device independent.
- Device dependent. It is possible to send a basic 9750 datastream to such a terminal. On the next #TREQ input, DC delivers the program a basic 9750 datastream. There is no longer device independence for the application. This is

only possible for DC-programs running through the DCAM line driver, not through UCF.

DC recognizes a basic 9750 datastream if the first byte is the header length byte as required by the VTSU in BS2000/OSD and if the second byte is a valid SOM (Start Of Message) for the 9750 terminal.

5.3 UCF Considerations

5.3.1 Generalities

All modules mentioned below are delivered on the installation tape. They were assembled using the macro defaults. If a module must be reassembled, include the BS2000/OSD system macrolib in the assembly JCL.

5.3.2 UCF-BATCH

UCF-BATCH allows you to run DC-tasks with output on SYSLST and input from the SYSDTA system file. It allows you also to redirect waiting CA-IDMS prints to SYSLST. Two UCF-Batch front-end modules are available: RHDCUBAT and RHDCUCFB.

RHDCUBAT: Allows you to run DC-tasks with output on SYSLST and input from the SYSDTA system file.

```
/START-PROG FROM-FILE=*MODULE(LIB=dba.loadlib,ELEM=RHDCUBAT,RUN-MODE=*ADV)
```

The module RHDCUBAT is compiled using the macro #UCFBTCH:

```
UCFBATCH #UCFBTCH UCFSYST=BATCH,      -
          OS=BS2K,PRINT=NO
```

RHDCUCFB: Allows you to redirect waiting CA-IDMS prints to SYSLST.

```
/START-PROG FROM-FILE=*MODULE(LIB=dba.loadlib,ELEM=RHDCUCFB,RUN-MODE=*ADV)
```

The module RHDCUCFB is compiled using the macro #UCFBTCH:

```
UCFBATCH #UCFBTCH UCFSYST=BATCH,      -
          OS=BS2K,PRINT=YES
```

5.3.3 UCF-TIAM

UCF-TIAM allows you to run DC-tasks on a terminal.

```
/CALL-PROCEDURE FROM-FILE=*LIB-ELEM(LIB=dba.source.lib,ELEM=FILE)
/START-PROG FROM-FILE=*MOD(LIB=dba.loadlib,ELEM=RHDCUCFT,RUN-MODE=*ADV)
```

Dynamic Logmode Selection: When the task switch 1 is set, RHDCUCFT prompts for the logmode. The logmode is entered with LOGMODE=lgm-value. Lgm-value must be a value defined in the logmode table (in BS2KDEVT). (See Chapter 6, “Data Communication Device Management (DCDM)” for more information.)

5.3.4 UCF-UTM

For a detailed description of the use of UTM, see the appropriate sections in the *CA-IDMS System Generation* and *CA-IDMS System Operations* manuals.

Currently, UTM does not support programs using CA-IDMS SQL access.

Note: When linking a UTM application, all includes of CA-IDMS modules should be from the CA-IDMS loadlib, except BSVCOPTS and BS2KDEVT that are in the DBA loadlib.

IDMSBSVC (CSECT in module IDMSBSVC) and IGC0 (CSECT in BSVCOPTS) have attribute RMODE=24. For UTM, this can be changed to RMODE=ANY during linkage. The link-edit parameters are dependent on how the program is linked.

- Linked with BINDER

```
//MODIFY-SYMBOL-ATTRIBUTES SYMBOL-NAME=IDMSBSVC,RES-MOD=ANY
//MODIFY-SYMBOL-ATTRIBUTES SYMBOL-NAME=IGC0,RES-MOD=ANY
```

- Linked with TSOSLNK

```
TRAITS IDMSBSVC,RMODE=ANY
TRAITS IGC0,RMODE=ANY
```

5.4 Program Loading

5.4.1 Library Search Chain

CA-IDMS system programs and DC user programs are loaded using a self-contained loader (IDMSLBS2). Only IDMSLBS2 is loaded via a BIND macro. For this reason, these FILE commands must be provided with an CA-IDMS batch program as well as with DC/CV:

```
/ADD-FILE-LINK LINK-NAME=CDMSLODR,F-NAME=idms.loadlib.idmslbs2
/ADD-FILE-LINK LINK-NAME=CDMSLIB,F-NAME=loadlib
/ADD-FILE-LINK LINK-NAME=CDMSLIBA,F-NAME=loadliba
:
/ADD-FILE-LINK LINK-NAME=CDMSLIBZ,F-NAME=loadlibz
/ADD-FILE-LINK LINK-NAME=CDMSLIB0,F-NAME=loadlib0
:
/ADD-FILE-LINK LINK-NAME=CDMSLIB9,F-NAME=loadlib9
```

Parameter	Description
<u>idms.loadlib.idmslbs2</u>	The filename of the library containing IDMSLBS2, usually the IDMS system loadlib.

Thus a maximum of 37 CA-IDMS/DC system and user load libraries can be assigned to CA-IDMS. It is not required to have a ADD-FILE-LINK command with LINK-NAME=CDMSLIB. If one load library is specified, the loader is able to load programs.

To ease of future installations and maintenance, avoid mixing system and user modules in the same library.

Note: In previous releases, linkname CDMSLIBS was used for =COPY IDMS. Since this linkname is included in the CDMSLIB chain, it should not be used for =COPY IDMS any more.

5.4.2 Notes for Central Version

- Central version requires an intermediate file for program loading. This intermediate file must be assigned to linkname CDMSPAM. Its size depends on the number of programs that are loaded by the central version. An absolute minimum allocation in PAM pages is (600,300).
- CA-IDMS/DC does not notice when a program is replaced in one of its load libraries; it continues to use the copy in its internal CDMSPAM file. To notify CA-IDMS/DC in such a case, use the DC master task function:

```
DCMT VARY PROGRAM program-name NEW COPY
```

- Test versions and libraries are supported as follows:

```
/ADD-FILE-LINK F-NAME=test.loadlib,LINK-NAME=Vnnnn
```

Parameter	Description
<u>test.loadlib</u>	Name of the test load library.
<u>nnnn</u>	The version number that is entered through the DCUF test facility.

5.4.3 Dynamic Subsystem Support

5.4.3.1 What is it?

The ability to put large parts of the CA-IDMS load modules in a BS2000/OSD subsystem. A subsystem in BS2000/OSD allows system-wide shareability of software. Please refer to the BS2000/OSD manuals for more details on DSSM.

5.4.3.2 Why Use Subsystems?

The advantages of using subsystems are:

- Less usage of real memory, because different CA-IDMS systems, CA-IDMS utilities and CA-IDMS user programs can all share the same copy of the modules in the subsystem.
- Less paging because of less real memory use.
- Elimination of program load times for all modules in a subsystem. This is a considerable saving for CA-IDMS batch programs or running IDMSDDDL or IDMSBCF with the EDT interface.
- Less address space because the BS2000/OSD tasks using modules of a subsystem all use the same address space.

5.4.3.3 When Not to Use Subsystems?

Each case should be investigated individually, but these factors must be considered:

- Modules that are modified often should not be put in a subsystem.
- A nucleus module residing in a subsystem cannot be reloaded unless a library is inserted into the CDMSLIB chain (with STGID) which comes BEFORE the subsystem library. If this is not done, various abends can occur, depending on the nucleus module(s) being reloaded.

- BS2000/OSD does not allow a program to disconnect from a subsystem; the program must terminate before its connection to the subsystem is broken. Therefore, if the CA-IDMS subsystem needs recycling, all central versions using the subsystem must be restarted. For this reason, sites for which 24x7 processing is important should not use the IDMS subsystem for their central versions.

5.4.3.4 Installing CA-IDMS Subsystems

Example: The DBA wishes to put all the modules of the MODLIST "IDMS150" into a Subsystem.

1. Allocate a library for the subsystem:

```
/CREATE-FILE $TSOSUSER.IDMS.SUBSYSTEM.LOADLIB.IDMS150X,SPACE=(999,99)
```

2. Execute the IDMSMOD function PREPSUBS:

```
/CALL-PROC NAME=*LIB-ELEM(LIB=dba.source.lib,ELEM=FILE)
/CALL-PROC NAME=*LIB-ELEM(LIB=dba.source.lib,ELEM=IDMSMOD), -
/ PROC-PAR=(PREPSUBS)
```

The procedure prompts for:

Parameter	Description
DSSMCAT	Filename of the SSCM source that is created. This source must be used by the BS2000/OSD system administrator to generate subsystem IDMS150X.
MODLIST	Member in the IDMS source library containing the list of modules to be put in the subsystem. IDMS150 is an existing member with modules that can be placed in the subsystem.
LIBPREF	Library prefix. In this example: \$TSOSUSER.
CDMSLX	Linkname CDMSLIBx that defines the position of the subsystem IDMS150X in the loadlib chain.

3. The procedure creates these files and modules:
 - The DSSMCAT file with the SSCM definitions for IDMS150X.
 - The module IDMS150X in the library \$TSOSUSER.IDMS.DSSM.LOADLIB.IDMS150X
 - Update the FILE procedure with:

```
/ADD-FILE-LINK F-NAME=$TSOSUSER.IDMS.DSSM.LOADLIB.IDMS150X, -  
/ LINK-NAME=CDMSLIBx
```

Note: The normal search algorithm also applies to subsystems. It is possible to use only some modules of a subsystem by issuing the appropriate ADD-FILE-LINK for CDMSLIBx commands.

4. Run SSCM to create the IDMS150X entry in the BS2000 subsystem catalog.
5. The BS2000 system administrator must issue:

```
/START-SUBSYSTEM IDMS150X
```

6. If the subsystem is not active when IDMSLBS2 tries to access it, the subsystem load module is loaded in private class 6 XS memory. As a result, there is a performance benefit, though less than with a real subsystem
7. To completely stop using the subsystem, release the CDMSLIBx linkname that identifies the subsystem load library.

5.4.3.5 Which Modules Can be Placed in a Subsystem?

Modules that are placed in a subsystem must satisfy these conditions:

- The module must be reentrant
- The module must not have direct references to the CA-IDMS CSA, because different CA-IDMS CV's and batch CA-IDMS programs may have their CSA at different addresses. Assembler programs with #BALI or linked with IDMSBALI and COBOL programs linked with IDMSCOBI contain direct CSA references, and should not be put in a subsystem.

The list of CA-IDMS modules eligible for a subsystem is contained in the member IDMS150 of the CA-IDMS source library. In addition to those modules, these module types can be put in a subsystem:

- Subschemas
- ADS dialogs
- ADS application load modules
- Maps
- IDD tables
- The shareable part of COBOL85 programs

5.5 Recovery from Program Errors

5.5.1 Trapping of Termination SVC

All the termination SVC's (TERM, TERMD, TERMJ, TRMJD) executed during the run of UCF/DC-Programs in user mode are trapped and substituted by an abend of the corresponding DC-TASK.

This allows the "RANGECHECK=TERM" option of the COBOL compiler to be used when compiling a COBOL UCF/DC-PROGRAM.

5.5.2 Recovery from Alignment Errors

Alignment errors (identified by interrupt weight IW=5C) are recovered automatically for UCF/DC- and Batch-programs in central mode and in local mode.

If this recovery potentially decreases system performance, you are notified by the creation of a file CDMSDUMP.yymmdd.tsn.

Parameter	Description
yymmdd	Date
tsn	The task sequence number

Please forward the printout of this file to your local Computer Associates or representative support.

5.6 Storage Protection

5.6.1 Description

To allow DC to switch storage keys, a special program is needed. If this program is not active and the CA-IDMS CV is generated with PROTECT, a warning message is issued; the storage protection feature is not activated. Storage protection is activated through BS2000/OSD subsystem management commands.

The value of the alternate user key is defined in SYSGEN. The default is 9.

5.6.2 Performance Considerations

When storage protection is active, one SVC is issued for all the memory pages allocated to the program, when switching from system to user mode and vice versa.

5.6.3 FETCH Protection

Fetch protection is not available.

5.6.4 Installation on BS2000/OSD OSD V2.0

Storage protection on BS2000/OSD OSD V2.0 is not supported.

5.6.5 Installation on BS2000/OSD OSD V3.0

The following description assumes the reader is familiar with BS2000/OSD documentation concerning system generation and management.

To install storage protection, execute these steps:

1. Under user ID TSOS, invoke the INSTSTPR procedure:

```
/CALL-PROC NAME=*LIB-ELEM(LIB=idms.source.lib,ELEM=INSTSTPR), -  
/ PROC-PAR=(BSLMSNAM=lms-program,IDMSSRC=idms.source.lib, -  
/ IDMSLOD=idms.loadlib,SSCM=sscm-compiler, STPRVER=120)
```

Parameter	Description
<u>lms-program</u>	Name of the LMS program, usually \$LMS
<u>idms.source.lib</u>	Name of the IDMS source library
<u>idms.loadlib</u>	Name of the IDMS load library
<u>sscm-compiler</u>	SDF command to invoke the SSCM compiler, usually START-SSCM

Successful installation creates:

- \$TSOS.SYSLNK.STPR.120 STPR subsystem loadlib
- \$TSOS.SYSREP.STPR.120 STPR subsystem repfile
- \$TSOS.SYSSSC.STPR.120 DSSM definitions for STPR subsystem

2. Add the STPR subsystem to the subsystem catalog using SSCM:

```
//ADD-CATALOG-ENTRY FROM-FILE=$TSOS.SYSSSC.STPR.120
```

3. Once the subsystem catalog is generated, the STPR subsystem can be made known to BS2000/OSD by the system administrator using the ADD-SUBSYSTEM command. For example,:

```
/ADD-SUBSYSTEM CATALOG=$TSOS.SYSSSC.STPR.120,TYPE=NEW-SUBSYS
```

4. Activate the STPR subsystem:

```
/START-SUBSYSTEM STPR
```

5.7 STGID Task

STGID is a task that is used to watch, monitor and tune the system. These are the possible parameters:

- HELP (default): display the HELP screen
- DISPLAY/LIST/PRINT: display, display and write to SYSLST or display and write a DC report, using these parameters:

Parameter	Description
ADSPACE	Address space layout
CDMSLIB	CDMSLIB through CDMSLIB9 load libraries assignments
DBIO	Database I/O statistics
ERE	ERE pool layout
KEPT	Kept storage id's
SCRATCH	Scratch area id's
TASK	All tasks invoked since startup

- CMD subt#/ALL BS2000-JCL-command
Enter a BS2000/OSD command.

Parameter	Description
<u>subt#</u>	Represents a subtask number: 0 for the MAINTASK, and 1, 2, etc. for the subtasks (only in a multi-tasking environment).
<u>BS2000-JCL-command</u>	A complete list of the commands that may be specified is contained in program BS2KCMDL. This program can be customized with the IDMSMOD function STGIDCMD and secured to fulfill your security needs.

- FREE storage-id

Parameter	Description
<u>storage-id</u>	Free kept storage by ID's

- VARY CDMSLIB_x ON/OFF:

Parameter	Description
<u>CDMSLIBx</u>	Include or exclude the specified library into or from the load library chain. CDMSLIBx may be any value from CDMSLIB to CDMSLIB9.

5.8 XS Support

Anywhere in the CA-IDMS documentation, notes referring to "XA" refer to 31-bit addressing, this is abbreviated as "XS" in the BS2000/OSD world.

Chapter 6. Data Communication Device Management (DCDM)

- 6.1 DCDM Introduction 6-3
 - 6.1.1 Presentation Layer 6-3
 - 6.1.2 Session Layer 6-3
 - 6.1.3 Transport Layer 6-5
- 6.2 DCDM Tables 6-6
 - 6.2.1 The Real Table of Devices 6-7
 - 6.2.2 The Logmode Table 6-7
 - 6.2.2.1 #BS2LGM Macro Parameters 6-8
 - 6.2.2.2 PRESENT Parameter Options 6-9
 - 6.2.2.3 PRESOPT Parameter Options 6-10
 - 6.2.2.4 TPROPT Parameter Options 6-11
 - 6.2.3 The PF-Key Table(s) 6-12
 - 6.2.4 The VTSU Control Block Table 6-12

6.1 DCDM Introduction

The management of data communication devices uses tables describing the presentation services, the session and transport options for all the devices, data communication access methods (DCAM, TIAM and UTM) and the DC-monitors (CA-IDMS/DC and IDMS/UCF).

The following is a brief overview of the functions and options of DCDM, classified in the Presentation Session and Transport Layer.

6.1.1 Presentation Layer

- A number of presentation options can be customized.
- Codes for the new line functions on printers can be customized.
- Mapping of Siemens K/F-keys to IBM PF-keys can be customized per application.
- All logical control characters are supported for printers, both in NATIVE and NON-NATIVE mode.
- All printers supported by VTSU are supported with buffering and logical acknowledgements.
- Printers automatically use the largest possible print buffer for a specific connection.

6.1.2 Session Layer

By default, CA-IDMS DC/UCF gets the device type identification from the access method (DCAM, TIAM). This default behavior can be overridden by forcing a LOGMODE.

Forcing a LOGMODE in DCAM is accomplished by exploiting the TRANSDATA connection message:

```
====>> PLEASE ENTER NET COMMAND
<<==== 0 IDMSDC,MSG=C'LOGMODE=SLINE'
```

Forcing a LOGMODE in TIAM can be done by setting task switch 1:

```
/MODIFY-JOB-SWITCHES ON=1
/START-PROG FROM-FILE=*MOD(LIB=dba.loadlib,ELEM=RHDCUCFT,RUN-MODE=*ADV)
```

In both cases, CA-IDMS DC/UCF uses the logmode entry, identified with "SLINE" regardless of the CA-IDMS DC/UCF or TRANSDATA network definition.

Note: Forcing a LOGMODE under UTM is not possible.

Using the Generalized SPOOL Interface with DC: BS2KGSPI is a DCAM program implementing the new SPOOL interface. Regardless of the CA-IDMS DC/UCF SYSGEN options, the LOGMODE-entry specified by BS2KGSPI is used, i.e. SSPOOL.

For CA-IDMS/DC and partially UCF-TIAM, some VTSU-editing options can be specified per logmode entry. If a complete control of VTSU is required, it is possible to force the system to use a specified VTSU control block; for example, specifying a coded character set name. For more information, refer to 6.2.2, “The Logmode Table” on page 6-7 and 6.2.4, “The VTSU Control Block Table” on page 6-12.

There is an alternative way of telling DC how to support a partner. To use this method, there must be a connection message of at least 7 bytes. The seventh byte must be a device-code that matches a BS2KDEV value in the real table of devices. The first six bytes should be either spaces or high-values. The spaces will request the usage of the HDX-protocol (see the TRPOPT-parameter in the logmode table). This alternative is supported for compatibility reasons and is primarily intended for DCAM-programs as DC-partners.

OMNIS Support: CA-IDMS/DC also supports OMNIS. To set up a session with CA-IDMS/DC, the following OMNIS-command or equivalent declaration must be given:

```
OPNCON pac,TYP=DCAM,PTNNAME=appl-id,LC=Y,LMSG='conn-msg'
```

Parameter	Description
<u>pac</u>	Partner code
<u>appl-id</u>	Application identifier of the CA-IDMS/DC system
<u>conn-msg</u>	Optional connection message can be used to specify a LOGMODE= <u>xxxxxx</u>

If no connection message is specified, CA-IDMS/DC looks in the real table of devices for an entry with device-code hexadecimal FF (BS2KDEV=X'FF'). With the BS2KDEVT module delivered during the installation, this corresponds to LOGMODE=S975X, and supports your OMNIS-terminal as a 975X terminal. The default can be changed by changing BS2KDEVT. Of course, you can write OMNIS-exits to specify LOGMODE=xxxxxx in the connection message to CA-IDMS/DC.

6.1.3 Transport Layer

- If specified, the system can use the "negative transport acknowledgements only" protocol. If appropriate to do so, this reduces both network and CPU-overhead.
- A connection message with the logmode entry used, can be sent to partners logging on to the system.
- The special protocol between CA-IDMS/DC and DCAM application programs can also be requested for connections, initiated from CA-IDMS/DC (ACQUIRE PTERMS).

6.2 DCDM Tables

The tables used by DCDM are all located in the module BS2KDEVT. At least three tables must exist:

- Real table of devices, identifying all device types known by DCAM, TIAM, UTM and CA-IDMS DC/UCF.

Each entry specifies the default logmode for the device. The default logmode is used when a specific logmode is not requested for a session (or connection).

- Table of logmodes: each entry describes presentation options, session options, transport options and access method parameters.
- One or more tables mapping Siemens K/F-keys to IBM's PF keys.

The standard installation provides you with source of the BS2KDEVT module that fulfils all your requirements in most cases. If required, a user-specific BS2KDEVT can be installed; it is generated by the IDMSMOD function: DCDM.

The following is an overview of the real device table entries installed by default:

PTERM Type	Model	Default LOGMODE	Description
D9750	1	S975X	975X display
D9003	1	S900X	900X type printer, supported physically
D9003	2	S900XBYP	900X type printer, connected in bypass, supported physically
DVTSU	1	SLINE	Line oriented terminal, VTSU logical control characters
DVTSU	4	SSCS	VTSU supported terminals, buffering provided by CA-IDMS DC/UCF, logical control characters
DVTSU	5	SSCSBYP	VTSU supported terminals, buffering provided by CA-IDMS DC/UCF, logical control characters, logical acknowledgements, (can be used for bypass-printers).
DVTSU	P/S/R/C	SLU62P, etc.	Siemens Emulated LU6.2

6.2.1 The Real Table of Devices

The real table of devices is generated by a set of #BS2DEV macros. #BS2DEV generates an entry per device type. There are five keyword parameters, that must be coded for each entry:

Parameter	Description
BS2KDEV	Device name as defined in the DCSTA TRANSDATA-macro. All possible values can be found in the listing of BS2KDEVT.
DCDEV	PTERM device type as known by CA-IDMS DC/UCF.
UTMDEV	UTM device code as known by UTM.
MODEL	The model of the PTERM type. Can be used to uniquely identify an entry in the real device table. The model by itself no longer defines a specific device characteristic as used to be in Release 10.0/10.1.
LOGMODE	Refers to an entry in the logmode table. The logmode determines how a device is supported.

How Does DCDM Search in this Table?: First of all, DCDM does not use this table if a LOGMODE is specified during connection setup. For information on specifying the LOGMODE, see 6.1.2, “Session Layer” on page 6-3.

When no logmode has been specified during session setup, the search algorithm depends on the TP-monitor used.

- For UCF-TIAM, the front-end uses the value of BS2KDEV, returned by TIAM, to search in the table. The logmode referred by the first matching entry is used.
- UCF-UTM is similar, but UTMDEV is used.
- With CA-IDMS/DC the search algorithm depends on the SYSGEN options. If a NAME clause is generated for a PTERM, the PTERM type and model of this PTERM is used to search in the table. If no matching NAME clause is generated for a terminal logging on to CA-IDMS/DC, the value of BS2KDEV as returned by DCAM is used.

6.2.2 The Logmode Table

This table contains different entries. Each entry describes the presentation services, the session and transport options for an CA-IDMS DC/UCF session. Each entry is generated by a #BS2LGM macro.

6.2.2.1 #BS2LGM Macro Parameters

Parameter	Description
label1	Defines the name of LOGMODE entry. Required
PRESENT1	Defines the type of presentation. See 6.2.2.2, “PRESENT Parameter Options” for available parameters.
PRESOPT	Defines a number of presentation options. See 6.2.2.3, “PRESOPT Parameter Options” for available parameters.
PFKTAB	Name of the table, containing the mapping between Siemens K/F-Keys and IBM PF-keys. Parameter is only meaningful for PRESENT=(975X/ADDS).
TRPOPT	Defines the transport options. See 6.2.2.4, “TPROPT Parameter Options” for available parameters.
PTETYPE	Default PTETYPE used when connecting with an explicit logmode, on a DCAM PTERM without NAME-clause.
PTEMODL	Default MODEL used when connecting with an explicit logmode, on a DCAM PTERM without NAME-clause.
CONRTRY	Number of times a connection retry is done for a printer. See Chapter 7, “Generalized Spool Interface” for more information.
LINELEN	Length of a line (defaults to 80).
PAGELEN	Length of a page, or number of lines on a screen (defaults to 24).
STXLEN	Number of characters in the STX-string.
STX	Hexadecimal STX-string; maximum length is 12.
ISTXLEN	Number of characters removed on input.
ETXLEN	Number of characters in the ETX-string.
NEWPAGE	Code for newpage control.
MAXLN	A non-zero value is the upper limit of the message length for devices with PRESENT=975X/SEM62. This parameter can be used if the maximum length returned by DCAM is not correct.
EDIT	CCB parameter. Refer to DCAM.
EDITIN	CCB parameter. Refer to DCAM.
EDITOUT	CCB parameter. Refer to DCAM.

Parameter	Description
PROC	CCB parameter. Refer to DCAM.
DEPROT	CCB parameter. Refer to DCAM.
ADRVTSU ₂	Specifies the address of the VTSU control block being used. The address must be a label of an entry in the VTSU control block table. See 6.2.4, "The VTSU Control Block Table." <ul style="list-style-type: none"> ■ If ADRVTSU is specified, the #BS2LGM parameters EDIT, EDITIN and EDITOUT are ignored. ■ If ADRVTSU is not specified, a VTSU control block is not used and the VTSU parameters in the #BS2LGM macro, EDIT, EDITIN and EDITOUT will control VTSU.

Note:

- 1 - Required
2 - Optional

Important Notes:

1. Changing the values for the parameters STXLEN, STX, ISTXLEN, ETXLEN, NEWPAGE, EDIT, EDITIN, EDITOUT, PROC and DEPROT requires knowledge and understanding of CA-IDMS DC/UCF, DCAM, TIAM and UTM.
2. There are many combinations of parameter values, devices and access methods, that do not work. If changes to the logmode are required it is a good practice to copy an existing one first.

6.2.2.2 PRESENT Parameter Options

Parameter	Description
USER	No presentation services required.
LINE	Device line oriented.
327X	327X - display.
975X	975X - display.
ADDS	ADDS - display.
SCS	Device is an SNA character string device. This presentation type is to be used for printers which are to be supported thru DCM-VTSU. Buffering by CA-IDMS DC/UCF is still effective.
328X	328X - printer.

Parameter	Description
900X	900X - printer.
PRESTEL	Videotex prestel.
TELETEL	Videotex teletel.
SEM62	Siemens Emulated LU6.2.

6.2.2.3 PRESOPT Parameter Options

PRESENT=975x

Parameter	Description
NCURINH	Cursor can be moved in protected fields.
CURINH	Cursor can not be moved.
NOVRLAP	Overlapping fields not allowed.
OVRLAP	Overlapping fields are allowed; this option should never be selected.
NMARK	Markable fields not supported.
MARK	Markable fields are supported. This option has some side effects.
NNOSKIP	NOSKIP fields are not supported.
NOSKIP	NOSKIP fields are supported. This option requires EXTATT, and has some side effects.
NIS2AP3	IS2 and IS3 do not always have to occur in pairs.
IS2AP3	IS2 and IS3 do always have to occur in pairs. The default option is better for 9750-terminals while IS2AP3 is more appropriate for 9755 terminals.
XNILLS	Binary zeroes and nills characters are translated to blanks in protected and NO-MDT fields.
NILLS	Nills are never translated.
NNILLS	Nills are always translated to blanks. This option has some side effects.
KBLOCK	Application correctly uses keyboard lock.
NKBLOCK	Application does not always correctly use keyboard lock.

PRESENT=327x/975x/ADDS

Parameter	Description
NEXTATT	Extended attributes, i.e. underlining, highlighting and color (only for 327X) are not available.
EXTATT	Extended attributes are available.

PRESENT=900x

Parameter	Description
CR	New line is coded as carriage return.
LF	New line is coded as line feed.
CR+LF	New line is coded as carriage return+line feed.
NFMFD	Formfeed is not supported.
FMFD	Formfeed is supported.
FMFDEND	Formfeed is supported. Beginning of report should not start with formfeed, but formfeed at end of report. This option is currently not implemented in printer driver (RHDCPRNT).

6.2.2.4 TPROPT Parameter Options

Parameter	Description
FDX	Standard message protocol.
HDX	Special message protocol, used for sessions between CA-IDMS/DC and DCAM-application program.
TACK	Transport acknowledgment required.
NTACK	Transport acknowledgment not required.
BYPASS	Bypass printer.
NBYPASS	Non-bypass printer.
CONMSG	Connection message allowed.
NCONMSG	Connection message not allowed.

When the HDX-protocol is used, each message sent by DC has a 1 byte header:

WRITEREQ	EQU	X'01'	write request
READREQ	EQR	X'20'	read request
WRTRDREQ	EQU	X'21'	tandem write/read
LASTMSG	EQU	X'81'	last msg of report
RPEREQ	EQU	X'40'	this is a report element. It starts at offset 4.

The HDX-protocol is used, regardless of TRPOPT=FDX, under these conditions:

- DC's partner is not a terminal.
- The connection message does not contain a LOGMODE parameter.
- There is a connection message of at least 7 bytes, with the first 6 bytes being spaces. The seventh byte must be a device-code that matches an entry in the real table of devices.

6.2.3 The PF-Key Table(s)

Each table defines a mapping between the Siemens K/F-keys and IBM's PF keys. The macros #BS2PFD and #BS2PFS must be used to generate the tables. Specifying a label for the #BS2PFD macros defines the name and the start of a new table.

Macro	Description
#BS2PFD	Defines the mapping between the Siemens key and the IBM-key if no PF key simulation is required. There are only 2 positional parameters. The first one is the Siemens K/F-key. The second one is the corresponding IBM-key.
#BS2PFS	Is similar, but must be used for PF-key simulation.

6.2.4 The VTSU Control Block Table

Each entry in this table defines a VTSU control block. A connection specifying a logmode that refers to a VTSU control block entry, uses VTSU with the specified parameters. VTSU parameter CCSNAME allows for the specification of a coded character set (CSS). For devices that support CCS, it is possible to specify a CCS that allows for the Euro (€) sign.

For more information on VTSU and coded character sets, refer to the BS2000/OSD manuals for DCAM, VTSU and XHCS.

Chapter 7. Generalized Spool Interface

- 7.1 Introduction 7-3
- 7.2 Concepts 7-4
- 7.3 Spool Interface Control Block (SPC) 7-5
- 7.4 Exit Routine for BS2KGSPI 7-6

7.1 Introduction

The Generalized Spool Interface of CA-IDMS provides these functions:

- Only one BS2000/OSD task is required to service all spoolout (PRINT and PUNCH) and spoolin (ENTER) requests from a specific central version. Different central versions must run different GSPI tasks.
- Specific customer requirements can be programmed in an exit to the generalized spool interface service task. This allows PRINT parameters (such as DEVICE, FORM, and SPACE) to be set depending on the CA-IDMS/DC parameters; for example, DESTINATION, USER, REPORT-ID, REPORTNAME, etc.
- All SPOOLOUT and SPOOLIN requests are logged with timestamp and CA-IDMS user-id.
- The transfer of CA-IDMS/DC spool queues to the spool service task performs better by blocking the spool records. A maximum buffer of 32KB is possible.
- The session or connection between CA-IDMS/DC and the spool service task does not have to be set up for each report. This results in a performance improvement. The connection stays active while the central version is running (NORELEASE option).
- The parameters for ENTER of a task can be defined on the LOGON of the ENTER-report created in CA-IDMS/DC.

7.2 Concepts

- Both spoolout (PRINT and PUNCH) and spoolin requests use the print support of CA-IDMS/DC. To route the requests to the BS2000/OSD spool interface task GSPI, the reports (eventually the DIRECT PRINTS) must be printed on a PTERM with a DCAM-NAME that matches the DCAM-application-name of the GSPI BS2000/OSD task.
- GSPI can process the reports depending on the report parameters. The report parameters are available in the Spool Interface Control Block (SPC), that is transferred to GSPI with each report.
- Typically, you would use the DESTINATION to differentiate PRINT, PUNCH and ENTER requests. Of course, the DESTINATION is also used to differentiate between different FORM or DEVICE parameters, or for ENTER's to differentiate between BS2000/OSD user IDs.
- The customer tailoring of PRINT, PUNCH and ENTER parameters or other BS2000 commands must be achieved by programming an exit. The exit can be written in any programming language supporting standard Siemens link conventions (i.e. COBOL, ASSEMBLER, etc.).

7.3 Spool Interface Control Block (SPC)

The SPC is a control block or datastructure transferred to the GSPI-task which contains the report parameters. It is described in Assembler by macro #GSPC. A COBOL-like description appears like this:

```

01 SPC
03 EYE      X(4) Eyecatcher containing the value "SPC"
03 STATUS  X(4) Function and status field, possible values are
              described in STATxxxx fields below.
03 QUEUENM X(16) Name of the queuename to be printed
03 REPRTNM X(8)  Name of the report to be printed
03 DESTNAT X(8)  Destination where report has been printed to
03 CLASS   X(2)  Class of report, displayable
03 CLASSNO S9(4) COMP Class of report
03 NODE    X(8)  Application name of GSPI, also the DCAM-name
              of the PTERM of the DC-printer.
03 PROC    X(8)  Processor name associated with NODE
03 DCNODE  X(8)  Application name of the DCAM-line controlling
              NODE
03 DCPROC  X(8)  Processor name associated with DCNODE
03 LTERM   X(8)  Logical terminal name of printer
03 PROGRAM X(8)  Name of program creating the report
03 REPRTID X(8)  Report id of this report
03 TOTLINE S9(5) COMP-3 Total number of lines of this report
03 COPYS   X      -
03 USER    X(32) User creating the report
03 TIME    X(11) Time of report creation
03 FILLER  REDEFINES TIME
05 HH      Hours of TIME
05 FILLER  :
05 MM      Minutes of TIME
05 FILLER  :
05 SS      Seconds of TIME
05 FILLER  :
05 FF      Hundredths of seconds of TIME
03 DATE    X(6)  Date of report creation (YY.DDD)
03 LINELEN S9(5) COMP Single line length
03 COPY 2  X      -
03 LOGMODE X(8)  Logmode to use for GSPI
03 ENTER   X(4)  Enter field. See below
03 SPACE   X      Space field. See below.
03 STATINIT X(4) "INIT" Startup of GSPI
03 STATEND X(4) "END " End of GSPI
03 STATEOR X(4) "EOR " End of report
03 STATBOR X(4) "BOR " Begin of report
03 STATNOPR X(4) "NOPR" Exit tells the calling program, BS2KGSPI,
              not to process record
03 STATPROC X(4) "0000" Spool record available
03 PREFIX  X(9)  See below
03 SUFFIX  X(20) See below

```

7.4 Exit Routine for BS2KGSPI

If required, an exit can be called thru BS2KGSPI. It must use the procedure GSPIX. The exit program is called:

- At the start of BS2KGSPI
- At the beginning of each report
- For each spool record
- At the end of each report
- At termination time of BS2KGSPI

A sample GSPIEXIT program is delivered in the CA-IDMS source library. Customization is accomplished by invoking IDMSMOD function GSPIEXIT. The source itself contains directions on how to customize.

Exit Program Requirements

- The exit program must have an entry-point name of GSPIEXIT, and must be linked with BS2KGSPI. The exit contains these parameters, in order:
 1. The SPC-control block
 2. The spool record in variable length format
 3. The COMMAND variable
- All parameters may be changed by the exit.
- The value of the STATUS variable in the SPC, controls both BS2KGSPI and GSPIEXIT.
 - STATUS="INIT" indicates the startup phase of BS2KGSPI. The exit is called to initialize SPC fields needed to establish the necessary DCAM environment (e.g. NODE) and to perform it's own initialization if necessary.
 - STATUS="BOR" indicates the beginning of the report. The exit is called with an empty SPOOL-record (length=4) and with COMMAND2 containing the proposed filename. COMMAND1 and COMMAND3 are set to blank.
 - The exit is called before the switch to command mode. If the exit sets the STATUS="NOPR", no switch to command mode occurs.
 - STATUS="0000" indicates a spool record is available.
 - If the exit sets the STATUS="NOPR", the record is skipped by BS2KGSPI.
 - STATUS="EOR", indicates the end of the report. The exit is called with an empty spool record (length=4). If parameters have been specified on the LOGON, those parameters (prefixed with a comma) are available in COMMAND3.

- COMMAND1 contains blanks. The exit is called before the switch to command mode. If the exit sets the STATUS="NOPR", no switch to command mode occurs.
- STATUS="END", indicates the end of the BS2KGSPI run. The exit should perform it's proper termination routines and return to BS2KGSPI.

Chapter 8. CA-IDMS/DC-DCAM Line Driver

- 8.1 Data Flow Control 8-3
- 8.2 Implementation of Connection Proposals 8-4
- 8.3 Passwords in DCAM 8-5
- 8.4 Transport Acknowledgments 8-6
- 8.5 Shareable DCAM Applications 8-7
- 8.6 Connection Retry for Printers 8-8
- 8.7 Support of ISO-DCAM 8-9

8.1 Data Flow Control

The DCAM line driver recognizes BCAM/DCAM resource bottlenecks on connections. An asynchronous wait for the DCAM GO-SIGNAL is done, without notifying the user or system administrator. As soon as the connection overload is rectified, the DCAM line driver is informed and data traffic resumes.

Note: The RPB used to set up the DCAM YSEND remains locked until the condition ends. Therefore, it may be necessary to increment the RPB COUNT parameter in the SYSGEN LINE statement.

8.2 Implementation of Connection Proposals

The DCAM line driver can accept connection proposals via the DCAM PROCON contingency as a result of the BS2000/OSD BCIN command. You must write an XKON macro in your PDN for each proposed station (defined with XSTAT) requiring a connection setup. For more information, please refer to relevant BS2000/OSD manuals.

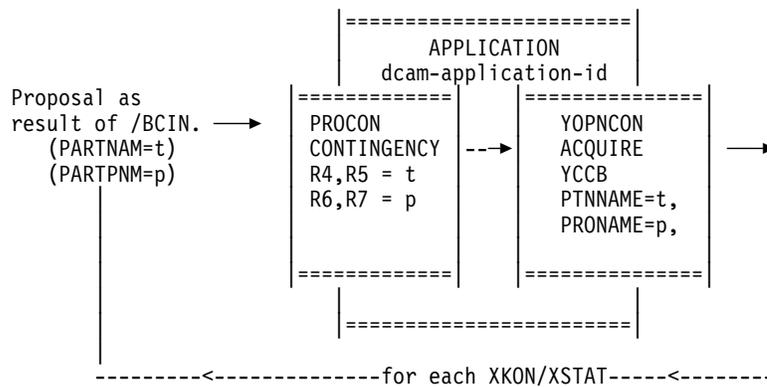
Example PDN

```

XSTAT  STATNUM= X
        STATNAM= ...

XKON   STATNUM= X
        PARTNAM= dcam-application-id
        PARTPNM= processor-of-partnam-a
...
    
```

Reaction of DCAM Driver



8.3 Passwords in DCAM

The following table shows you the possibilities concerning the use of application and connection passwords.

Connection	O <u>appl-id</u> , PW='password'
Sysgen	ADD LINE IS <u>line-id</u> APPLICATION IS <u>appl-id</u> USER PASSWORD IS <u>password</u> .
Startup JCL	/SET-DCAM-APPLICATION-LINK APPL-NAME= <u>appl-id</u> , / LINK-NAME= <u>line-id</u> , / PROT=(CON-PASS= <u>log-pass</u> ,SHARE-PASS= <u>pass-1</u>), / SHARE-PASS= <u>pass-2</u>
BCAM-RDF	XSTAT APPNAM= <u>appl-id</u> , APPPW= <u>pass-2</u>
DCAM <u>driver</u>	YACB APPNAME= <u>appl-id</u> , LINK= <u>line-id</u> , LOGPASS= <u>password</u> , USEPASS=, USEPW=
#PLEDS	PLEID, PLE5DAID, PLE5APSW Refer to the <i>BS2000/OSD Command Language</i> and <i>BS2000/PDN</i> manuals for more information.

8.4 Transport Acknowledgments

The DCAM line driver supports the parameter PROTOCOL IS DEFRESP/EXPRES. If a negative acknowledgment is received with EXCEPTION RESPONSE protocol, the used PTE is set out of service.

Please refer to the *CA-IDMS System Generation* manual.

8.5 Shareable DCAM Applications

The DCAM line driver supports multi-tasking, but no more than eight BS2000/OSD TSN's can run because of the current BS2000/OSD/DCAM limits.

8.6 Connection Retry for Printers

If the VARY-IN of a printer connected to DC fails, the new retry facility tries to prevent the associated PTERM from being set out of service. This facility is only active under these circumstances:

- The VARY-IN is requested by the printer task
- The VARY-IN fails because the printer is already acquired by another system, including BS2000/OSD, or it is already acquired by another PTERM within the same DC system
- The corresponding logmode entry contains this parameter:

```
CONRTRY=(time-value,counter-value)
```

Parameter	Description
<u>time-value</u>	Represents the time, in seconds, the system waits before retrying to connect the printer. If the specified value is 0, the system will not retry to connect the printer; its associated PTERM is set out of service. The maximum value is 255.
<u>counter-value</u>	Represents the number of times the system tries to connect the printer. If the specified value is 0, the system tries indefinitely, every <u>time-value</u> seconds.

Note: The standard device table specifies CONRTRY=(60,10) for all printers.

8.7 Support of ISO-DCAM

CA-IDMS/DC also supports DCAM lines with ISO protocol instead of the classical NEA protocol. This allows data communication based on the transport services as described by ISO.

Requirements and/or limitations:

- Because DCAM (ISO) is used, all limitations of DCAM (ISO) apply to the ISO support in CA-IDMS/DC.
- Partners making transport connections with the ISO line can not use NEA-services that are not available in ISO. The important services are:

Parameter	Description
TACK	Transport acknowledgments
EDIT	No editing services are provided
SYSCODE	Translation services not available
LOGPW	LOGON password not supported
PTNCHAR	DCAM does not supply any info about the partner
SUBGROUP	Subgroup structure of message not supported

- The transport option HDX (Half-Duplex) in the logmode is currently not supported, because TACKS are used to inform partners to do a read, even with TRPOPT=(NTACK) in the logmode.
- The MDATA=Y service (More Data) is currently not available.

In practice, the requirements and limitations mean:

- Terminals cannot directly connect to the ISO line. This does not mean that presentation services for terminals are not presented, i.e. one can write a DCAM(ISO) program that does make a transport connection with the ISO line, requests presentation services for a 9750 terminal from DC and does the terminal I/O using TIAM.
- Ensure the logmode for the transport connection does not contain TRPOPT=(TACK,HDX). Other invalid parameters are ignored by DCAM (ISO).

To generate a DCAM(ISO) line in CA-IDMS/DC, a new parameter has been added to the LINE entity definition. Two different values may be assigned to this parameter:

- DCAM TYPE=ISO: allows multiple sessions between two partners
- DCAM TYPE=NEA: use the NEA transport function of DCAM. This is the default value.

The PROTOCOL (DEFRESP/EXPRES) and the USER PASSWORD parameters from a DCAMLIN type line may not be specified for a DCAM-ISO line.

Chapter 9. APPC in the BS2000/OSD environment

9.1 Overview of APPC	9-3
9.1.1 Terminology	9-3
9.1.2 How it Works	9-4
9.1.3 LU6.2 Verbs	9-5
9.1.4 Example of Conversation	9-5
9.1.5 CA-IDMS and APPC	9-7
9.2 SEM62 Implementation	9-9
9.2.1 Overview	9-9
9.2.2 Different Kinds of Sessions	9-9
9.2.3 Multiple Session Management	9-10
9.3 SEM62 Generation	9-11
9.3.1 Device Table	9-11
9.3.2 System Generation	9-12
9.3.3 LU-name Identification	9-13
9.4 DCMT LU Command	9-19

9.1 Overview of APPC

Advanced Program-to-Program Communication (APPC), is an IBM standard that allows distributed or cooperative processing. It enables two programs, eventually on different processors, to communicate over a conversation. APPC is composed of logical and physical definitions of the system network. The logical component is the LU6.2 protocol, that defines the rules governing the exchange of information between the two programs.

In a BS2000/OSD environment, the LU6.2 protocol is not supported by TRANSDATA. To allow one application to communicate with another application running under a different CA-IDMS/DC system, this protocol is emulated. The Siemens emulation of LU6.2 is implemented, and is referred to as SEM62. The goal of SEM62 is to allow two CA-ADS dialogs or programs, on different CA-IDMS/DC systems, eventually on different processors, to communicate between each other using the LU6.2 protocol.

SEM62 is fully compatible with LU6.2 protocol at program level.

9.1.1 Terminology

The following terms are frequently used in the IBM (SNA) environment. For more information and details, please refer to:

- NAU (Network Addressable Unit): Each element in an SNA network to which data can be sent. Examples are the PU's and the LU's.
- PU (Physical Unit): Portion of a device (usually programming or circuitry or both) that performs control functions for the device in which it is located, and eventually, for other devices attached.

Examples of PU's:

- PU2.0: 3274-controllers, "dumb" terminals
- PU2.1: intelligent controllers, allow 'PEER-TO-PEER'
- PU4.0: front-ends and communication controllers
- PU5.0: host processors

- LU (Logical Unit): Device or program by which an end user (a program or an operator) gains access to the network. It is the source of all requests coming into the network.

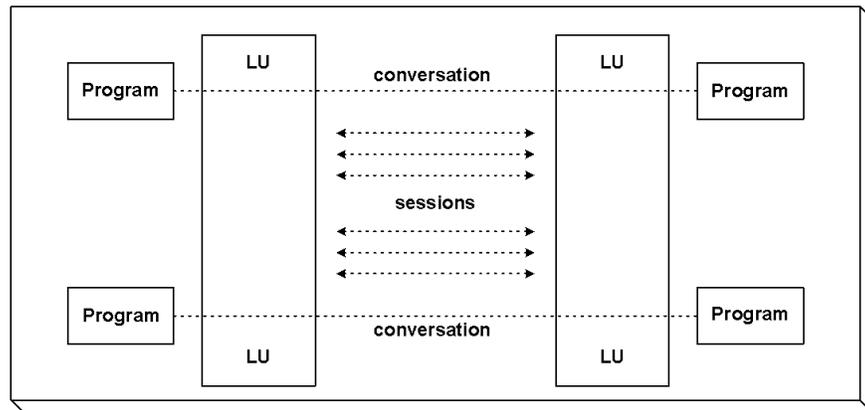
Examples of LU's:

- LU0: no protocol
- LU1: SCS printers (comparable to Siemens VTSU)
- LU2: 3270 terminals
- LU6.2: programs

In the DCAM environment, logical units are commonly named partners.

Session VS Conversation

- **SESSION:** Logical connection between two logical units. Two logical units that are connected by a single physical connection, can have one or many sessions between them. A terminal connected with CA-IDMS/DC through the VTAM line driver is an example of a session.
- **CONVERSATION:** Set of transactions between two programs. A conversation is typically a short-term connection over a session. One session carries one conversation at a time.



9.1.2 How it Works

The LU6.2 protocol structures a program-to-program conversation like a polite conversation: one side talks and the other side listens.

- One program (called the primary program) starts the conversation by calling the other program (the secondary program).
- The conversation goes back and forth, with the current speaker (in send state) always in control:
 - When the listener (in receive state) wants to speak, the listener requests permission.
 - The speaker must grant permission before the program can reverse roles. If the speaker approves the request, the listener switches to send state and the speaker to receive state.
- The two programs agree on the rules governing the conversation at the outset. For example, if they allow confirmation, when the speaker issues a CONFIRM, the listener responds with CONFIRMED, then the conversation continues.

9.1.3 LU6.2 Verbs

This list describes the LU6.2 verbs that can be used in CA-ADS dialogs. For more information, please refer to the appropriate CA-ADS manual.

Verb	Definition
ALLOCATE	Begins a conversation with a secondary dialog
CONTROL SESSION	Agrees to the rules governing the conversation started by the first dialog (extension to LU6.2)
DEALLOCATE	Ends the conversation
SEND-DATA	Sends data to the remote program
RECEIVE-AND-WAIT	Waits for a response from the remote program
CONFIRM	Sends a confirmation request to the remote program
CONFIRMED	Sends a confirmation reply to the remote program
PREPARE-TO-RECEIVE	Switch local side to receive state
REQUEST-TO-SEND	Receiver wants to change direction
SEND-ERROR	Notifies the other partner of an application-level error

Note: The equivalent functionality is available for each verb on the #TREQ macro level.

9.1.4 Example of Conversation

An example of a conversation between two CA-ADS dialogs is provided. This information is given under the form of input to the DDDL compiler, and is divided in three parts:

Records Definition

```

ADD RECORD APPCRC.
  02 APPCRCCO      PIC 9(4)  USAGE IS COMP.
  02 APPCR CER     PIC 9(4)  USAGE IS COMP.
  02 APPCR CRS    PIC 9(4)  USAGE IS COMP.
  02 APPCR CWR    PIC X(18) USAGE IS DISPLAY.

ADD RECORD APPCREC1.
  02 APPCEL11     PIC X(50)  USAGE IS DISPLAY.
  02 APPCEL12     PIC 9(8)   USAGE IS COMP.

ADD RECORD APPCREC2.
  02 APPCEL21     PIC X(50)  USAGE IS DISPLAY.
  02 APPCEL22     PIC 9(8)   USAGE IS COMP.

```

Allocator Dialog

```
ADD PROCESS APPCPM1 MODULE SOURCE FOLLOWS
  ALLOCATE LU-NAME 'LUNAME1' TPN 'APPCDIA2' SYNC-LEVEL CONFIRM NOFORMAT.
  IF APPCCODE LT 0 THEN CALL ERREXIT.

  MOVE 'APPCDIA1: DATA1' TO APPCEL11.
  MOVE 1                    TO APPCEL12.
  SEND-DATA APPCREC1.
  IF APPCCODE LT 0 THEN CALL ERREXIT.

  RECEIVE-AND-WAIT APPCREC1.
  IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'DATA-COMPLETE'
    THEN CALL ERREXIT.

  RECEIVE-AND-WAIT.
  IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'SEND' THEN CALL ERREXIT.

  MOVE 'APPCDIA1: DATA3' TO APPCEL11.
  MOVE 3                    TO APPCEL12.
  SEND-DATA APPCREC1.
  IF APPCCODE LT 0 THEN CALL ERREXIT.

  DEALLOCATE SYNC-LEVEL.
  IF APPCCODE LT 0 THEN CALL ERREXIT.
  LEAVE ADS.

  DEFINE SUBROUTINE ERREXIT.
    WRITE TO LOG TEXT 'APPCDIA1: ***** APPC ERROR *****'.
    MOVE APPCCODE          TO APPCRCCO.
    MOVE APPCERC           TO APPCRCER.
    MOVE REQUEST-TO-SEND-RECEIVED TO APPCRCRS.
    MOVE WHAT-RECEIVED    TO APPCRCWR.
    SNAP RECORD (APPCRC).
    IF NOT (APPCCODE=5 AND APPCERC=1) THEN DEALLOCATE ABEND.
    LEAVE ADS.
  GOBACK.
MSEND.
```

Allocatee Dialog

```

ADD PROCESS APPCPM2 MODULE SOURCE FOLLOWS
CONTROL SESSION NOFORMAT.
IF APPCCODE LT 0 THEN CALL ERREXIT.

RECEIVE-AND-WAIT APPCREC2.
IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'DATA-COMPLETE'
  THEN CALL ERREXIT.

RECEIVE-AND-WAIT.
IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'SEND' THEN CALL ERREXIT.

MOVE 'APPCDIA2: DATA2' TO APPCEL21.
MOVE 2                  TO APPCEL22.
SEND-DATA APPCREC2.
IF APPCCODE LT 0 THEN CALL ERREXIT.

RECEIVE-AND-WAIT APPCREC2.
IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'DATA-COMPLETE'
  THEN CALL ERREXIT.

RECEIVE-AND-WAIT.
IF APPCCODE LT 0 OR WHAT-RECEIVED NE 'CONFIRM-DEALLOCATE'
  THEN CALL ERREXIT.

CONFIRMED.
IF APPCCODE LT 0 THEN CALL ERREXIT.

DEALLOCATE LOCAL.
IF APPCCODE LT 0 THEN CALL ERREXIT.
LEAVE ADS.

DEFINE SUBROUTINE ERREXIT.
  WRITE TO LOG TEXT 'APPCDIA2: ***** APPC ERROR *****'.
  MOVE APPCCODE      TO APPCRCCO.
  MOVE APPCERC       TO APPCR CER.
  MOVE REQUEST-TO-SEND-RECEIVED TO APPCR CRS.
  MOVE WHAT-RECEIVED TO APPCR CWR.
  SNAP RECORD (APPCRC).
  IF NOT(APPCCODE=5 AND APPCERC=1) THEN DEALLOCATE ABEND.
  LEAVE ADS.
GOBACK.
MSEND.

```

9.1.5 CA-IDMS and APPC

The CA-IDMS/APPCC provides APPC support for these scenarios:

Scenario 1: Emulated APPC through CA-ADS, within the same central

version: Two CA-ADS dialogs, within the same CA-IDMS/DC system, communicate using the emulated LU6.2 line driver (RHDCD0FI), by specifying these parameters on the ALLOCATE statement:

- LU-NAME=OWN
- MODE-NAME=RHDCD0FI

Scenario 2: Real APPC through CA-ADS: Two CA-ADS dialogs, on different CA-IDMS/DC systems, on different processors, communicate using the SNA line driver (RHDCD0EV).

Scenario 3: Real APPC through PCs: This is the same functionality than offered by scenario 1, but the communication uses the real LU6.2 protocol (IBM only).

9.2 SEM62 Implementation

9.2.1 Overview

SEM62 is also using the emulated LU6.2 line driver (RHDCD0FI). When the SEM62 interface detects a CA-ADS dialog issuing an ALLOCATE command that refers to a logical unit name different from OWN, SEM62 sends the request to the target LU using a transport medium composed of two DCAM lines connected together, one at each side of the conversation. The same session between the two DCAM lines is used for the transport of all the other requests of the conversation between the two logical units.

Because SEM62 needs the DCAM line driver to transport the requests to the partner through the network, SEM62 requires the product CA-IDMS/DC.

Within an CA-IDMS/DC system, a logical unit is associated with a DCAM line. The logical unit name is equal to the DCAM application-id defined at sysgen, or taken from a APPLICATION command at open line time.

CA-IDMS/DC allows multiple (parallel) sessions with other logical units. A session is represented by a physical terminal element (PTE) for which no LTE is associated. Multiple sessions are defined at sysgen by defining multiple PTE's with the same dcam-node-name, i.e. the same PTE name-clause. (Later in the document, such a PTE is called a SEM62 PTE).

The maximum number of sessions between LU's is limited by the number of PTE's. The number of sessions that are available at anytime is limited by the number of PTE's that are in service.

9.2.2 Different Kinds of Sessions

In the real SNA environment, a session viewed from one of the logical units may be contention winner or contention loser: contention winner on one side means contention loser on the other side, and vice versa. This notion is used to avoid conflict when two logical units try to allocate a conversation on the same session at the same time: in this case, the side with contention winner gets first priority.

This feature has no equivalent in a BS2000/OSD (DCAM) environment; therefore, it is simulated this way: from a given logical unit (LU1), there are two types of session possible with another logical unit (LU2):

- ALLOCATOR or PRIMARY sessions — These sessions can be used by a logical unit (LU1) to allocate a conversation with the remote logical unit (LU2)
- ALLOCATEE or SECONDARY sessions — These sessions can only be allocated by the remote logical unit (LU2) when the partner starts the conversation

So, in the Siemens Emulated LU6.2 environment, we say a logical unit may have two different models, a primary or a secondary model:

- A primary model of a logical unit is connected to its partner via a primary session
- A secondary model of a logical unit is connected to its partner via a secondary session

At sysgen level, all PTE's associated with primary sessions have the parameter MODEL=P; the parameter MODEL=S is assigned to all PTE's associated with secondary sessions.

9.2.3 Multiple Session Management

To manage the multiple sessions between two LU's, a multiple session service manager CNOS (Change Number Of Sessions) is delivered with the CA-IDMS/APPC product. This program is activated by the DCMT VARY LU command to alter the number of sessions between two LU's, and to alter the characteristics of these sessions. (See the DCMT command later in this chapter).

These CNOS requests are negotiated on a dedicated conversation. For this purpose, two special sessions should be reserved between two logical units: one primary and one secondary session. The primary session uses a PTE with MODEL=R, while the secondary session uses a PTE with MODEL=C.

These two new models R and C are reserved for CNOS; it is recommended to generate such a couple of PTE's (model R and C) for each target logical unit.

Note: In absence of dedicated CNOS PTE, a standard SEM62 PTE, if available, is used to carry out the CNOS request to the target LU.

Different MODE Names: In the real SNA environment, different sessions between logical units may have different classes of service and even different protocols. This is accomplished via the MODEENT parameter; generated on a VTAM PTE, and refers a specific MODEENT entry in the VTAM mode table.

At the user level, a specific MODEENT is selected via the MODE-NAME parameter on the ALLOCATE command. If the user wants to select the emulation mode, for example for a conversation between two logical units belonging to the same central version, it must specify the parameter MODE-NAME=RHDCD0FI.

In the BS2000/OSD environment, only the emulation mode is available. The MODE-NAME=RHDCD0FI parameter is forced for each ALLOCATE command executed from a user application, whatever be the value specified by the user.

There is one exception to this rule: to use its reserved conversation, the multiple session manager CNOS always issues its ALLOCATE command with the parameter MODE-NAME=SNASVCMG. If no dedicated conversation is reserved for CNOS, or if the first ALLOCATE command has failed, CNOS will retry with MODE-NAME=RHDCD0FI.

9.3 SEM62 Generation

This paragraph describes the modifications to the CA-IDMS/DC system to run APPC in a BS2000/OSD environment. First, a description of the new logmode entries from the device table is given, followed by a description of the sysgen parameters for specifying the LINE and PTERM entity. Finally, some considerations allowing a greater independence between the names of the LU's and the actual network names are given.

9.3.1 Device Table

SEM62 requires four new entries in the real table of devices, and four new logmode entries in the logmode table, and consists of:

*	#BS2DEV BS2KDEV=STADHOST,DCDEV=PTEDVTSU,MODEL=P, LOGMODE=SLU62P,UTMDEV=00	-
*	#BS2DEV BS2KDEV=STADHOST,DCDEV=PTEDVTSU,MODEL=S, LOGMODE=SLU62S,UTMDEV=00	-
*	#BS2DEV BS2KDEV=STADHOST,DCDEV=PTEDVTSU,MODEL=R, LOGMODE=SLU62R,UTMDEV=00	-
*	#BS2DEV BS2KDEV=STADHOST,DCDEV=PTEDVTSU,MODEL=C, LOGMODE=SLU62C,UTMDEV=00	-
SLU62P	#BS2LGM PRESENT=SEM62,EDIT=USER,PROC=(BINARY,SIGNAL), TRPOPT=(NTACK),PTETYPE=PTEDVTSU,PTEMODL=S, STXLEN=2,ETXLEN=1,MAXLN=4000	- -
SLU62S	#BS2LGM PRESENT=SEM62,EDIT=USER,PROC=(BINARY,SIGNAL), TRPOPT=(NTACK),PTETYPE=PTEDVTSU,PTEMODL=P, STXLEN=2,ETXLEN=1,MAXLN=4000	- -
SLU62R	#BS2LGM PRESENT=SEM62,EDIT=USER,PROC=(BINARY,SIGNAL), TRPOPT=(NTACK),PTETYPE=PTEDVTSU,PTEMODL=C, STXLEN=2,ETXLEN=1,MAXLN=256	- -
SLU62C	#BS2LGM PRESENT=SEM62,EDIT=USER,PROC=(BINARY,SIGNAL), TRPOPT=(NTACK),PTETYPE=PTEDVTSU,PTEMODL=R, STXLEN=2,ETXLEN=1,MAXLN=256	- -
*		

The default BS2KDEVT device table delivered with the CA-IDMS software contains these new entries.

The same value must be assigned to the MAXLN parameter defined in the SLU62P and SLU62S logmodes for all central versions which communicate via APPC (the same applies for the SLU62R and SLU62C logmodes).

9.3.2 System Generation

To run SEM62 properly, this information must be sysgen'd in the system dictionary:

- One special DCAM line for each logical unit that is declared in this system (the local system)
- One D0FI line that is used for all kinds of LU6.2 emulation:
 - On the same central version using the LU-NAME=OWN parameter on the ALLOCATE command
 - On different central versions using SEM62

Enough PTE's must be generated in the D0FI line to satisfy the maximum number of concurrent requests the system will support.

- One special program: RHDCCNOS, and one special task: 06F1. These are used by the multiple session manager.

The correct definition of this special program and task, and an example of D0FI line generation is provided. It is followed by a description of the LINE and PTERM parameters associated with a SEM62 DCAM line.

CNOS and D0FI Line Definition

```
PROGRAM RHDCCNOS LAN=ASS PROG NOPROTECT REEN .
TASK 06F1 INVOKES RHDCCNOS NOINPUT PRIORITY=240 INACTIVE INTERVAL OFF.

LINE D0FILINE TYPE LAPPCEMU.
PTERM D0FIPT01 TYPE PAPPCEMU.
LTERM D0FILT01 .
PTERM D0FIPT02 TYPE PAPPCEMU.
LTERM D0FILT02 .
PTERM D0FIPT03 TYPE PAPPCEMU.
LTERM D0FILT03 .
PTERM D0FIPT04 TYPE PAPPCEMU.
LTERM D0FILT04 .
PTERM D0FIPT05 TYPE PAPPCEMU.
LTERM D0FILT05 .
PTERM D0FIPT06 TYPE PAPPCEMU.
LTERM D0FILT06 .
PTERM D0FIPT07 TYPE PAPPCEMU.
LTERM D0FILT07 .
PTERM D0FIPT08 TYPE PAPPCEMU.
LTERM D0FILT08 .
PTERM D0FIPT09 TYPE PAPPCEMU.
LTERM D0FILT09 .
PTERM D0FIPT10 TYPE PAPPCEMU.
LTERM D0FILT10 .
```

SEM62 LINE Definition: In a SEM62 line, the APPLICATION IDENTIFICATION parameter represents the name of the logical unit in the local system. This value can be overridden by a APPLICATION command in the startup JCL. See 9.3.3, "LU-name Identification" for more information.

A new parameter has been added to the LINE entity definition. Two different values may be assigned to this parameter:

- DCAM TYPE=ISO: allows multiple sessions between two partners.
- DCAM TYPE=NEA: to use the NEA transport function of DCAM. This is the default value.

All the other parameters associated with a DCAMLIN type line apply also for a SEM62 line, except the PROTOCOL (DEFRESP/EXPRES) and the USER PASSWORD parameters. This is a restriction of DCAM-ISO.

SEM62 PTERM Definition: A SEM62 PTERM is characterized by the following:

- It has no associated LTERM
- The TYPE is equal to DVTSU
- The MODEL is equal to P, S, R or C
- The NAME-clause contains the name of the target logical unit. This value can be overridden by a CONNECTION command in the startup JCL. See 9.3.3, “LU-name Identification” for more information.
- Its PROCESSOR NAME-clause contains the name of the processor where the target logical unit resides. A reserved value *LUNAME* may be assigned to this parameter to allow a greater independence with the network names. See 9.3.3, “LU-name Identification” for more information.

Two other options on the PTERM definition may be used for SEM62, with the following meaning:

- ACQUIRE/NOACQUIRE: when ACQUIRE is specified on a SEM62 PTE, the session with the partner is established when the line is opened; in the other case, the session is activated at ALLOCATE time.

The ACQUIRE option may be specified on any four type of PTE models. In any way, it should be specified for the PTERM's associated with the reserved CNOS sessions.

- RELEASE/NORELEASE: when RELEASE is specified on a SEM62 PTE, the session is automatically disconnected during the DEALLOCATE processing.

If the user wants to change the number of sessions between logical units, use the DCMT VARY LU command instead of the RELEASE option.

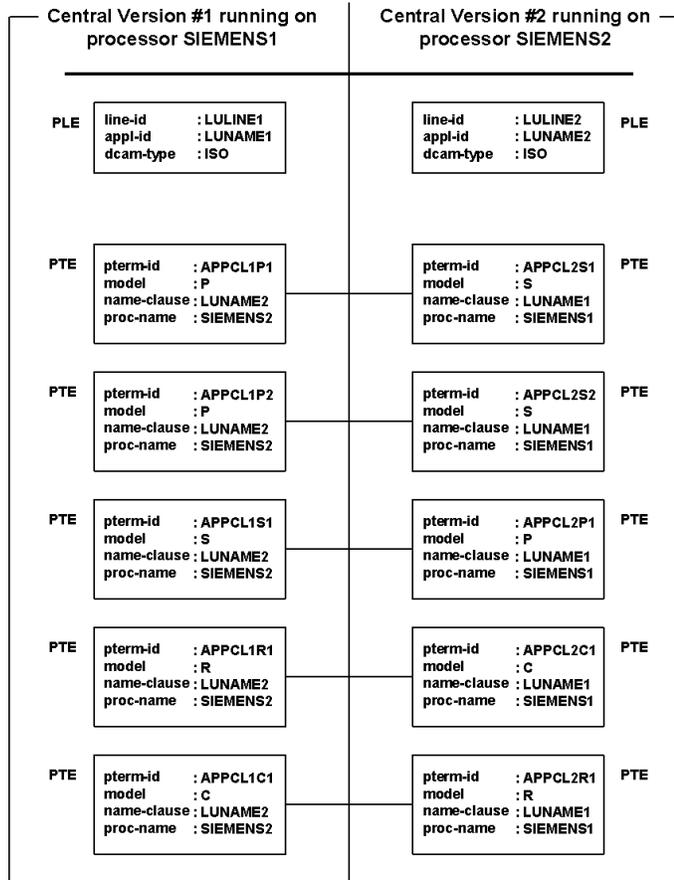
9.3.3 LU-name Identification

There are two ways to identify the LU names in the system: statically or dynamically.

Static Identification: This is the case where the logical unit names are pre-generated using the sysgen compiler. The APPLICATION ID of one SEM62 DCAM line corresponds to the PTERM NAME-clauses in the SEM62 line associated with the partner. An example of this configuration and its sysgen input are provided.

Example of Configuration Defined Statically

Example of Configuration Defined Statistically



Sysgen Compiler Input

```

LINE LULINE1 TYPE=DCAMLIN APPLICATION ID=LUNAME1
DCAM TYPE=ISO RPB COUNT=5.
PTERM APPCL1P1 IN LINE LUNAME1 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME2 PROCESSOR NAME=SIEMENS2.
PTERM APPCL1P2 IN LINE LUNAME1 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME2 PROCESSOR NAME=SIEMENS2.
PTERM APPCL1S1 IN LINE LUNAME1 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME2 PROCESSOR NAME=SIEMENS2.
PTERM APPCL1R1 IN LINE LUNAME1 TYPE=DVTSU MODEL=R ACQUIRE
NAME=LUNAME2 PROCESSOR NAME=SIEMENS2.
PTERM APPCL1C1 IN LINE LUNAME1 TYPE=DVTSU MODEL=C ACQUIRE
NAME=LUNAME2 PROCESSOR NAME=SIEMENS2.

LINE LULINE2 TYPE=DCAMLIN APPLICATION ID=LUNAME2
DCAM TYPE=ISO RPB COUNT=5.
PTERM APPCL2P1 IN LINE LUNAME2 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME1 PROCESSOR NAME=SIEMENS1.
PTERM APPCL2S1 IN LINE LUNAME2 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME1 PROCESSOR NAME=SIEMENS1.
PTERM APPCL2S2 IN LINE LUNAME2 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME1 PROCESSOR NAME=SIEMENS1.
PTERM APPCL2R1 IN LINE LUNAME2 TYPE=DVTSU MODEL=R ACQUIRE
NAME=LUNAME1 PROCESSOR NAME=SIEMENS1.
PTERM APPCL2C1 IN LINE LUNAME2 TYPE=DVTSU MODEL=C ACQUIRE
NAME=LUNAME1 PROCESSOR NAME=SIEMENS1.

```

Dynamic Identification: To provide a greater independence between the TRANSDATA network names and the associated fields in PLE's and PTE's, DCAM provides the SET-DCAM-APPLICATION-LINK and SET-DCAM-CONNECTION-LINK commands.

The SET-DCAM-APPLICATION-LINK command redefines the APPLICATION-ID associated with a DCAM line in a DC system, without the need to run the sysgen compiler. Its syntax consists of the following:

```

/SET-DCAM-APPLICATION-LINK LINK-NAME=dc-line-id, -
/ APPL-NAME=new-dcam-appl-id

```

Parameter	Description
<u>dc-line-id</u>	Line-id specified in the sysgen LINE statement
<u>new-dcam-appl-id</u>	New application-id that is really used for the DCAM line

The SET-DCAM-CONNECTION-LINK command redefines the name of the communication partner that is connected with a DCAM line in the same or in another DC system. Its syntax is the following:

```

/SET-DCAM-CONNECTION-LINK LINK-NAME=pte-name-clause,           -
/ PARTNER-ADDRESS=(PARTNER-NAME= new-comm-partner-name,       -
/ PRONAME=processor-name)

```

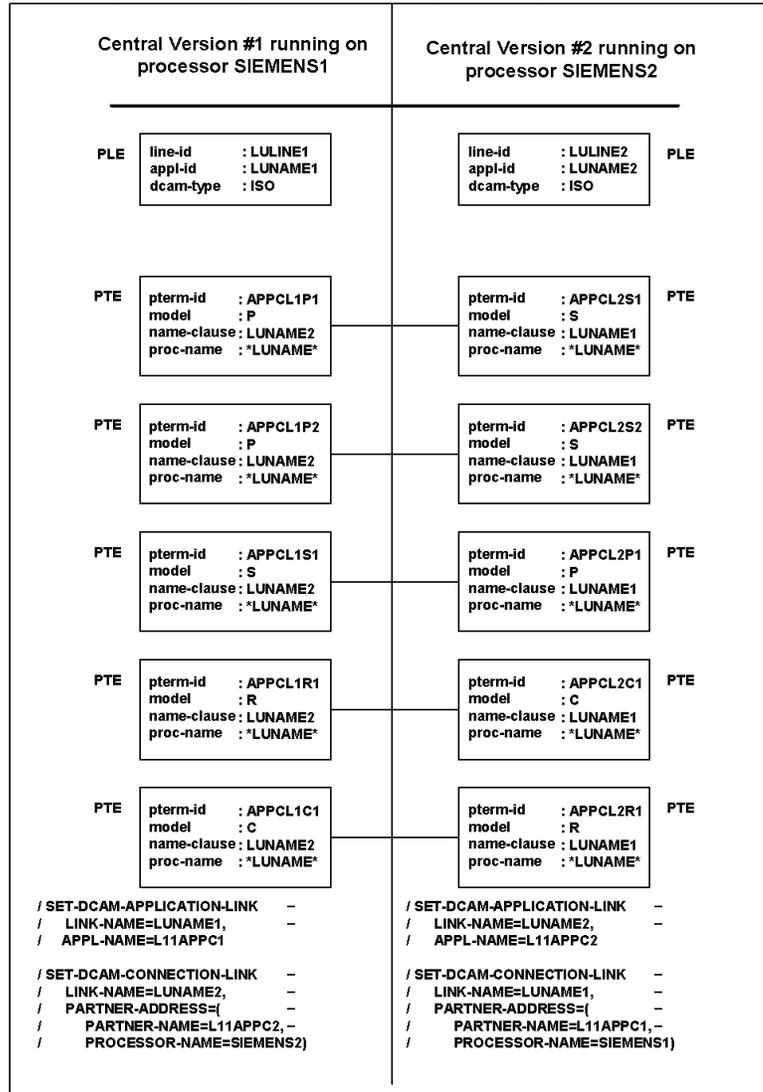
Parameter	Description
<u>pte-name-clause</u>	DCAM node name of the partner, defined in the NAME clause of the sysgen PTERM statement
<u>new-comm-partner-name</u>	New name used for the connection
<u>processor-name</u>	Name of the processor on which the communication partner is defined

To be able to use this method of dynamic LU name identification, the following modifications need to be done in the sysgen compiler input:

- The processor name has to be defined to *LUNAME*
- The same logical name must be specified for the PTE NAME-clause in the allocator system, and for the PLE line-id in the allocatee system.

An example of configuration between two systems is given hereafter, followed by the equivalent input to the sysgen compiler.

Example of Configuration Defined Dynamically
Example of Configuration Defined Dynamically



Sysgen Compiler Input

```
LINE LUNAME1 TYPE=DCAMLIN APPLICATION ID=LUNAME1
DCAM TYPE=ISO RPB COUNT=5.
PTERM APPCL1P1 IN LINE LUNAME1 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME2 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL1P2 IN LINE LUNAME1 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME2 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL1S1 IN LINE LUNAME1 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME2 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL1R1 IN LINE LUNAME1 TYPE=DVTSU MODEL=R ACQUIRE
NAME=LUNAME2 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL1C1 IN LINE LUNAME1 TYPE=DVTSU MODEL=C ACQUIRE
NAME=LUNAME2 PROCESSOR NAME='*LUNAME*'.

LINE LUNAME2 TYPE=DCAMLIN APPLICATION ID=LUNAME2
DCAM TYPE=ISO RPB COUNT=5.
PTERM APPCL2P1 IN LINE LUNAME2 TYPE=DVTSU MODEL=P ACQUIRE
NAME=LUNAME1 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL2S1 IN LINE LUNAME2 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME1 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL2S2 IN LINE LUNAME2 TYPE=DVTSU MODEL=S ACQUIRE
NAME=LUNAME1 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL2R1 IN LINE LUNAME2 TYPE=DVTSU MODEL=R ACQUIRE
NAME=LUNAME1 PROCESSOR NAME='*LUNAME*'.
PTERM APPCL2C1 IN LINE LUNAME2 TYPE=DVTSU MODEL=C ACQUIRE
NAME=LUNAME1 PROCESSOR NAME='*LUNAME*'.

```

9.4 DCMT LU Command

DCMT LU commands apply to SEM62 physical terminals. They differ slightly from the commands which apply to SNA physical terminals in an IBM environment.

Two actions are available for DCMT LU commands: DISPLAY and VARY. For more information, see *CA-IDMS System Tasks and Operator Commands*.

Chapter 10. BS2000/OSD Services

10.1 Introduction	10-3
10.1.1 Overview	10-3
10.1.2 Terminology	10-4
10.1.3 Assumptions and Restrictions	10-5
10.2 Installation	10-6
10.2.1 Contents of the Product	10-6
10.3 Making Use of BS2000/OSD Services	10-7
10.3.1 Step 1 — Define the Service	10-7
10.3.2 Step 2 — Generate the Service	10-8
10.3.3 Step 3 — Link Your Application Program(s)	10-9
10.3.4 Step 4 — Create an ENTER File for the Service	10-9
10.3.5 Step 5 — Define the Service to CA-IDMS/DC	10-9
10.4 Exits to Program BS2KSRVB	10-10
10.4.1 SRVSTART	10-10
10.4.2 SRVPREP	10-10
10.4.3 SRVPOSTP	10-10
10.4.4 SRVSTOP	10-10
10.4.5 SRVABEND	10-10
10.5 Administration Program BS2KSRVA	10-11
10.5.1 Functionality	10-11
10.5.2 Syntax	10-11
10.6 Abend Codes	10-12
10.7 Example: Service for DMS Access	10-13

10.1 Introduction

10.1.1 Overview

Applications running under CA-IDMS/DC might want to make use of BS2000/OSD functionality or other products running on BS2000/OSD. Examples are LEASY, SESAM, UDS, DCAM applications and DMS file access.

Previously, writing such applications was difficult, because:

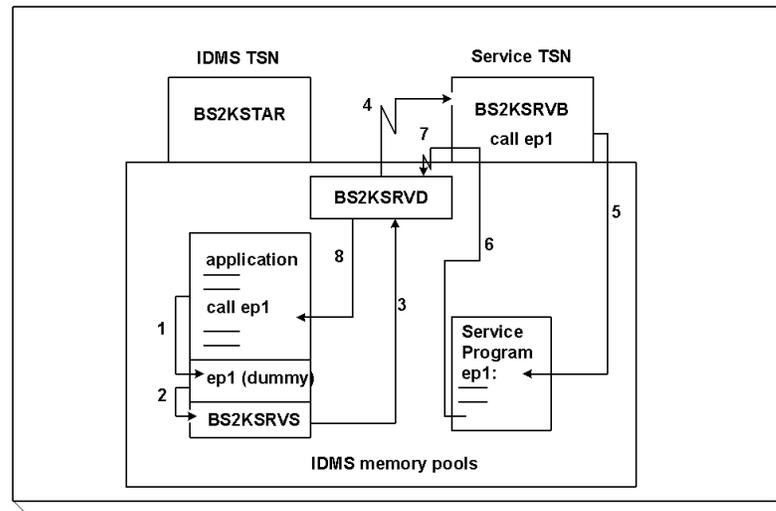
- BS2000/OSD products have their own conventions that are often incompatible with CA-IDMS/DC. Also, they use SVC's that are not allowed under CA-IDMS/DC; for example, abend detection (STXIT), storage management (REQM), loader information (LINK, VSVI, ITABL).
- Using BS2000/OSD functionality can cause performance problems and even abends of the central version:

Examples:

- In case of an I/O, the whole CA-IDMS region is put in an operating system wait state until the I/O finishes.
- A DMS abend of such an application program leads to an abend of the IDMS central version.

The goal of BS2000/OSD Services is to eliminate the above mentioned difficulties. This can be done only if a "boundary" exists between the CA-IDMS/DC application program and the functionality it wants to use. This boundary is the "call interface" for products like LEASY, SESAM and UDS; for DCAM applications and file access thru DMS, the boundary has to be taken care of in the program design. Technically, a boundary is one (or more) entry point(s).

Principally, the application is cut on the boundary. The part above the boundary continues to execute under CA-IDMS/DC, the other part, which we call the service module, is still be loaded in the CA-IDMS/DC region, but execute in a separate (batch) TSN that shares the CA-IDMS/DC address space. A front- and back-end is needed to allow communication between CA-IDMS/DC and the batch TSN.



The program flow then becomes: The application program does a call to an entry point. If not yet done, the service module containing the real entry point is loaded in the CA-IDMS/DC address space. Control is not given to the real entry, but the BS2000/OSD task assigned to process the call for the named entry is posted. Next the interface module waits upon completion of the call.

The posted BS2000/OSD task, using the same address space as CA-IDMS/DC, gives control to the real entry point. After completion of the call, it posts the CA-IDMS/DC task that issued the request.

The advantages of this approach are:

1. The restrictions applying under CA-IDMS/DC dispose.
2. All I/O and waits occur in the BS2000/OSD task. This means other CA-IDMS/DC tasks can be processed.
3. Eventual abends that can not be trapped (e.g. DMS error DMS09A1) result in:
 - Abend of the BS2000/OSD TSN
 - Abend of the requesting CA-IDMS/DC task

The CA-IDMS central version will not abend.

10.1.2 Terminology

SERVICE: A service is identified by a name (srvname), which must be unique within the CA-IDMS/DC environment.

ENTRY: An entry is identified by a name (entry-name), which must be unique within a service. entry-name must be identical to the name of the entry point being called.

Examples:

- For LEASY, entry-name should be LEASY
- For UDS, entry-name should be DML

10.1.3 Assumptions and Restrictions

1. The BS2000/OSD service TSN **must** run under the same user ID as CA-IDMS/DC. This restriction might be removed in a later version of the product.
 2. The address space available to the BS2000/OSD service TSN is equal to the maximum available address space for the user ID minus the CA-IDMS/DC address space.
 3. Standard operating system call conventions are assumed. This means:
 - R13 points to an 18 fullword save area
 - R14 contains the return address
 - R15 contains the address of the entry point to call
- Note:** These restrictions are automatically fulfilled if the application program is written in COBOL.
4. No IDMS DC nor IDMS DB calls are allowed in a service. This restriction might be removed in a later version of the product.
 5. A call to an entry must not return addresses of storage outside the CA-IDMS/DC address space. Since COBOL does not support pointers, this restriction is only meaningful for Assembler programs.
 6. If the CA-IDMS central version runs with storage protection, application programs using BS2000/OSD Services **must** run unprotected.
 7. A service must be defined to CA-IDMS/DC. This means it must be defined in sysgen or added via a DCMT command.
 8. Communication with a BS2000/OSD TSN is single threaded. This means that while a call to an entry is being processed by a BS2000/OSD TSN, no CA-IDMS/DC program can make use of the executing BS2000/OSD TSN. Single threading is handled by the product.

It is possible to assign multiple BS2000/OSD TSN's to a service. When doing so, additional restrictions apply since it is not guaranteed that the same application program will always use the same BS2000/OSD TSN. This means a call to an entry must not rely on previous calls.

A simple example might clarify this:

Assume a service with 2 BS2000/OSD TSN's defined for file access: TSN1 and TSN2. One application asks to open a file. Control is passed to TSN1 for opening the file. While this process is busy, a second service call comes in to open another file. TSN2 processes the call. After completion of the 2 requests, a call comes for a read on the second file. Since TSN1 is free, it is posted to process the request. However, since the file was opened in TSN2, this might lead to problems.

10.2 Installation

BS2000/OSD Services is a separate orderable product. The installation is performed by adding the following to your IJMP file:

CA-IDMS/BS2KSERV INSTALL

10.2.1 Contents of the Product

1. Procedure GENSERV. Procedure is installed in the CA-IDMS DBA source library.
2. Load modules BS2KSRVA, BS2KSRVB, BS2KSRVD and BS2KSRVS. These modules are installed in the CA-IDMS load library.
 - BS2KSRVA is the administration program.
 - BS2KSRVB and BS2KSRVD are the programs that communicate respectively in the batch and the CA-IDMS/DC TSN's.
 - BS2KSRVS is to be linked with the application program that does a call to an entry.
3. Macros #SDM and #SRVPL. These macros are installed in the CA-IDMS macro library. They are used by the GENSERV procedure.
4. Source members DLODSRVS, DLODSRVM. These members are installed in the CA-IDMS source library.
 - DLODSRVS contains the SYSGEN input.
 - DLODSRVM contains messages to be added to the message area.

10.3 Making Use of BS2000/OSD Services

Before you are able to use a BS2000/OSD service, you have to execute these steps.

10.3.1 Step 1 — Define the Service

Defining a service consists of creating an LMS source member with version DEF in the CA-IDMS DBA source library. The member must contain service definition statements, which are in free format input with a few restrictions:

- Code only one statement on a line
- A statement must be completed on one line, i.e. continuation is not permitted
- Comments are only allowed in comment lines, not on real statement lines

Syntax for service definition statements:

* Comment

All lines starting with an asterisk in column 1 are treated as comments.

SERVICE-NAME= srvname

SERVICE-NAME

This required statement must be the first non-comment line in the service definition.

srvname

A freely selectable name that must be different from the entry point name(s) within the service. It is used for:

- The jobname of the BS2000/OSD TSN(s) associated with a service.
- The name of the load module containing both the service definition and the real entry points defined in the SRVEP statement.

Maximum length of srvname is 8.

MAXTSN= maxtsn#

maxtsn#

The number of TSN's defined for the service. Default value is 1.

Note: For each TSN defined in a service, an ERE is used for communication. Check if enough ERE's are defined in your CA-IDMS/DC system.

MAXABND= maxabnd#

maxabnd#

The number of times a service is automatically restarted after it abended. Default value is 0, i.e. no automatic restart.

ENTER=YES/NO

ENTER

Specifies whether the service should be (re)started by CA-IDMS

YES

CA-IDMS (re)starts the service. This is the default. These two parameters are relevant:

ENTER-FILE-NAME=enter-filename

ENTER-PARMS='enter-parms'

IDMS starts the service TSN as follows:

/ENTER enter-filename.enter-parms

enter-parms

Must be enclosed in single quotes.

NO

CA-IDMS does not start the service.

SRVEP= (ep1name[,...])

SRVEP

A required statement and must be the last in the service definition.

ep1name

The name of an entry point within a service. At least one entry point should be defined. The total number of entry points that can be defined is limited due to the maximum length of an SRVEP line: 55 characters.

10.3.2 Step 2 — Generate the Service

Call the GENSERV procedure as follows:

```
/CALL-PROC NAME=*LIB-ELEM(LIB=dba.sorc.lib,ELEM=IDMSMOD), -
/ PROC-PAR=(GENSERV,MEMBER=member,LIBPREF=libpref)
```

Parameter	Description
<u>member</u>	Name chosen for the LMS source member with version DEF created in the previous step.
<u>libpref</u>	The prefix used to create a PLAM library named <u>libprefLIB.member</u> . This prefix can be chosen freely.

If the procedure is called interactively, member is displayed on the screen. If it does not exist, a blank EDT screen is displayed on which you can fill in the service definition. After having adapted or created the service definition, press K1 (or type HALT and press the Enter key) to resume processing.

The procedure creates 2 sources named SRVEPDEF and member (meaning of member: see above) in the above mentioned PLAM library and assembles them into the same library.

Then, the service module link member is displayed on the screen (it is created if it does not exist yet; add the necessary BINDER statements to complete it). Eventually adapt it and again press K1 (or type HALT and press the Enter key). The service load module is linked into the CA-IDMS DBA load library.

After successful execution of the procedure, the message "Generation of service defined in member successfully completed." should appear.

If an error occurs during execution of the procedure, the message "Error occurred during procedure run." is displayed. Check the syslst to find out the error cause.

10.3.3 Step 3 — Link Your Application Program(s)

Every application program doing a call to a service entry must include:

- SRVEPDEF from the PLAM library mentioned in the previous step
- BS2KSRVS from the CA-IDMS load library
- IDMSBALI from the CA-IDMS load library

10.3.4 Step 4 — Create an ENTER File for the Service

The ENTER file should contain all necessary JCL to execute service requests (e.g. FILE assignments, eventual preparation steps, etc.) followed by:

```
/START-PROG FROM-FILE=*MOD(LIB=idms.loadlib,ELEM=BS2KSRVB,RUN-MODE=*ADV)
```

Parameter	Description
<u>idms.loadlib</u>	The name of your CA-IDMS load library.

10.3.5 Step 5 — Define the Service to CA-IDMS/DC

Defining the service means making the service module known to CA-IDMS/DC. This can be done in two ways:

1. Statically in SYSGEN
2. Dynamically via DCMT VARY DYNAMIC PROGRAM

Important: The program **must** be defined as NONOverlayable and without storage protection.

10.4 Exits to Program BS2KSRVB

Program BS2KSRVB has the following exits:

1. Startup exit SRVSTART
2. Service preprocessing exit SRVPREP
3. Service postprocessing exit SRVPOSTP
4. Normal termination exit SRVSTOP
5. Abend exit SRVABEND

Upon entry of an exit:

- R14 contains the address to return to
- R15 contains A(exit)

All registers may be destroyed by the exit.

10.4.1 SRVSTART

This exit receives control during startup of a BS2000/OSD service TSN. No parameters are passed.

10.4.2 SRVPREP

This exit receives control prior to a call to a service entry. R13 points to an 18 fullword save area in which, at offset 16 (decimal), registers R15 thru R12 that are used for calling the service entry can be found.

10.4.3 SRVPOSTP

This exit receives control after a call to a service entry. R13 points to an 18 fullword save area in which, at offset 16 (decimal), registers R15 thru R12 as they were returned by the called service entry can be found.

10.4.4 SRVSTOP

This exit receives control upon normal termination of a BS2000/OSD service TSN. No parameters are passed.

10.4.5 SRVABEND

This exit receives control upon abnormal termination of a BS2000/OSD service TSN. No parameters are passed.

Note: SRVABEND does not get control when the TSN is being canceled. If this situation must be trapped, define a STXIT in the SRVSTART exit.

10.5 Administration Program BS2KSRVA

The administration program BS2KSRVA is by default invoked by task code SRVADMIN.

10.5.1 Functionality

1. Display information about one or all service(s).
2. Stop one or all service(s).

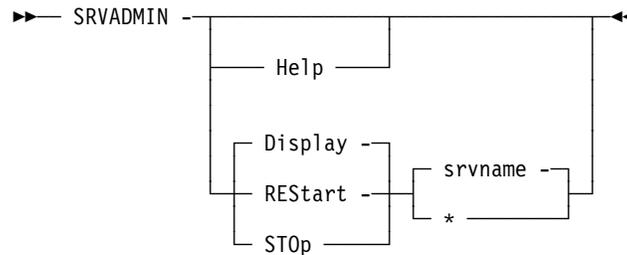
This means that any request to an entry within the specified service(s) results in a "service not available" abend. Also, the BS2000/OSD service TSN(s) are terminated normally. The ERE(s) used by the service(s) remain in use.

3. Restart one or all service(s).

This means the ERE(s) used by the specified service(s) are freed. The first call to an entry within a service results in a restart of the related service.

4. Syntactical help can be displayed.

10.5.2 Syntax



10.6 Abend Codes

All abend codes are in the message dictionary and come from module BS2KSRVD.

DCMT D MES ABNDxxxx (xxxx is the abend code)

This gives you more information about an abend message. Their description can be found in Appendix B, “Error Messages and Abend Codes” on page B-1.

10.7 Example: Service for DMS Access

Currently an application APPLI1 that needs information from a BS2000/OSD file does calls to two entry points, DMSEP1 and DMSEP2, within module DMSMOD to access the file, thereby putting the whole CA-IDMS central version in an operating system wait state during I/O. Goal is to create a service that will process the I/O.

Following are the different steps to execute:

1. Verify that all assumptions and restrictions that apply to BS2000/OSD Services are fulfilled.
2. Call the GENSERV procedure:

```
/CALL-PROC NAME=*LIB-ELEM(LIB=dba.source.lib,ELEM=IDMSMOD), -
/ PROC-PAR=(GENSERV, MEMBER=DMSSERV, LIBPREF=TMP.)
```

The service definition member is shown in EDT full screen. The first time the procedure is called, type the service definition when receiving a blank EDT screen:

```
SERVICE-NAME=DMSSERV
MAXTSN=1
MAXABND=2
ENTER=YES
ENTER-FILE-NAME=ENTER.DMSSERV.TEST
ENTER-PARMS='TIME=199,JOB-CLASS=JCBATCH'
SRVEP=(DMSEP1,DMSEP2)
```

Press K1 to proceed. After two assemblies, the service link member is shown in EDT full screen. The first time the procedure is called, add the following statement after the REMARK statements:

```
//INCLUDE-MODULE ELEM=DMSMOD,LIB=application.lib
```

Parameter	Description
<u>application.lib</u>	The library where DMSMOD resides.

Eventually add RESOLVE-BY-AUTOLINK statements. Press K1 to proceed. The module DMSSERV is linked in the CA-IDMS DBA loadlib.

3. Adapt the link of APPLI1:
Change

```
//INCLUDE-MODULE ELEM=DMSMOD,LIB=application.lib
```

To

```
//INCLUDE-MOD ELEM=SRVEPDEF,LIB=TMP.LIB.DMSSERV  
//INCLUDE-MOD ELEM=BS2KSRVS,LIB=idms.loadlib
```

Relink APPLI1.

4. Create the ENTER file ENTER.DMSSERV.TEST:

```
/SET-LOGON-PARAMETERS  
/ASSIGN-SYSOUT ...  
/ASSIGN-SYSLST ...  
/CREATE-FILE ...  
/ADD-FILE-LINK ...  
/  
:  
/START-PROG FROM-FILE=*MOD(LIB=idms.loadlib,ELEM=BS2KSRVB,RUN-MODE=*ADV)  
/LOGOFF NOSPOOL
```

5. Define the service to CA-IDMS/DC:

```
DCMT Vary Dynamic Program DMSSERV ASsembler  
NONOverlayable NOProtect .
```

Now the BS2000/OSD service DMSSERV is ready for use.

Chapter 11. CA-IDMS/Server

- 11.1 Introduction 11-3
- 11.2 CAICCI BS2000/OSD Operation 11-4
 - 11.2.1 System Requirements 11-4
 - 11.2.2 Description 11-4
- 11.3 CAICCI Task 11-5
 - 11.3.1 CAICCI Input Parameters 11-6
 - 11.3.1.1 Input Parameter Descriptions 11-6
 - 11.3.2 CAICCI Control Commands 11-8
 - 11.3.2.1 CAICCI Command Descriptions 11-10
- 11.4 CCITCP Task 11-11
 - 11.4.1 CCITCP Input Parameters 11-12
 - 11.4.1.1 CCITCP Input Parameter Descriptions 11-12
 - 11.4.2 CCITCP Control Commands 11-14
 - 11.4.2.1 CCITCP Command Description 11-15
- 11.5 IDMS Task 11-17
 - 11.5.1 BS2KSTAR Input Parameters 11-17

11.1 Introduction

The CA-IDMS/Server product is available for CA-IDMS users running on the BS2000/OSD operating system.

This product delivers PC-Windows client/server access to CA-IDMS mainframe databases in accordance with the Microsoft Open Database Connectivity (ODBC) protocol. This capability allows a user running CA-Visual Express on a PC to access data stored in a CA-IDMS database on the mainframe.

The communication between the PC and the mainframe is based on CAICCI, the Common Communication Interface from Computer Associates. The CAICCI service provides support for BS2000/OSD-to-PC connectivity through TCP/IP.

The following sections describe how CAICCI and its TCP/IP feature are implemented on BS2000/OSD and how they can be started and controlled.

For information on how to install and setup the CA-IDMS/Server and CAICCI software on the PC, please refer to the following manuals:

- *CA-IDMS Server User Guide (Release 4.2)*
- *Unicenter TNG Framework for OS/390 Administrator Guide*

Please note that from the PC side no specific changes are needed, and no specific action has to be taken to access a BS2000/OSD system.

11.2 CAICCI BS2000/OSD Operation

11.2.1 System Requirements

In order to run CA-IDMS/Server on BS2000/OSD, the socket runtime library, usually cataloged under \$TSOS as \$TSOS.SYSLNK.SOCKETS.010, should contain 2 load modules, YSOCLIB and YSOTU@, having both a version equal to 103 or higher

11.2.2 Description

The BS2000/OSD implementation of CAICCI and its TCP/IP feature allows users running CA-Visual Express (or any other ODBC applications) on a PC connected to the mainframe through TCP/IP, to access data stored on a host CA-IDMS database.

Communication between two partners is performed through a CCI memory pool. On one side, the CCI request issued by the PC is transferred to a CCITCP driver running on the mainframe, using the socket programming interface. The CCITCP driver executes the CCI request on behalf of the PC, locally on the mainframe.

On the other side, the CA-IDMS database system, through its CCI defined line, receives and processes the request, and answers the PC via the CCITCP driver.

The CCITCP driver is able to serve different PC's concurrently, that may access the same or different CA-IDMS systems. Also, there may be different CCITCP drivers running on the same host. In the socket terminology, one CCITCP driver corresponds to one PORT number.

The CCI memory pool is initiated and controlled by one BS2000/OSD task, that is called the CAICCI task.

Each CCITCP driver also corresponds to one BS2000/OSD task. We call it the CCITCP task.

Finally, a CA-IDMS system is controlled by one or more BS2000/OSD tasks (when running in multi-tasking mode).

Let's examine how these different tasks can be started, and how they can be controlled.

11.3 CAICCI Task

The CAICCI task is responsible for the initialization and control of the whole CCI environment. When starting the task, these functions are executed:

- Enabling and initiating the CCI memory pool
- Loading the CCI interface module into the CCI memory pool, so the same copy of the program is used by all the CCI users. The CCI interface module contains a memory manager that manages all the space within the CCI memory pool.

Once the CAICCI task is started, it is able to process commands that can be entered via the INFORM-PROGRAM command, or a job variable. See 11.3.1, “CAICCI Input Parameters” for a complete description of these commands.

The CAICCI task is also responsible for the control of the CCI memory pool. At regular intervals, the CAICCI task gets control and scans the CCI memory pool in order to free all resources that are not used, resulting from task abends, PC reboots or any other abnormal terminations.

The CAICCI task is not invoked by the CCI users, the CCITCP task or a CA-IDMS system and it is not CPU intensive; therefore, it does not need to be started with a high priority.

Please note that when the CAICCI task is restarted after an abend, the whole CCI memory pool is reinitialized. This means before restarting the task, one should ensure that all CCI users that were active at the time, have released their CCI resources; for CA-IDMS, this means the CCI line must be varied offline. The CCITCP drivers must also be stopped. See 11.4, “CCITCP Task” for more information.

The CCI.LIB library contains the E-CAICCI JCL member that is used to start the CAICCI task. Please note that the TASKLIB system file must be assigned to the same library.

To allow commands to be entered through a job variable, a user job variable must be defined in the startup JCL of the CAICCI task. The linkname associated with this job variable must be *JCAICCI.

```

./CAICCI SET-LOGON-PARAMETERS
/MODIFY-JOB-OPTION LOGGING=(INFORMATION-LEVEL=MEDIUM)
/MODIFY-TEST-OPTIONS DUMP=YES
/SET-JV-LINK JV-NAME=jv.name1, LINK-NAME=*JCAICCI 1
/ASSIGN-SYSOUT TO=L.CAICCI.OUT
/ASSIGN-SYSDTA TO=*SYSCMD
/SET-TASKLIB LIB=CCI.LIB
/START-PROG FROM-FILE=*MOD(ELEM=CAICCI, LIB=CCI.LIB, RUN-MODE=*ADV)
enter your input parameters
END
/LOGOFF NOSPOOL

```

1 This command is optional.

11.3.1 CAICCI Input Parameters

The CAICCI program accepts input cards from SYSDTA with these limitations and characteristics:

- Only one parameter per input card is allowed.
- All input cards are free format.
- The input cards are composed of the parameter keyword followed by an equal sign and the parameter value.

This table lists all the valid parameters and their default values:

Keyword	Description	Default
CCIMPNAM	CCI memory pool name	CCIMP010
CCIMPADR	CCI memory pool address	Lowest XS address
CCIMPSIZ	CCI memory pool size	16M
CCI#TSN	Maximum number of TSN's on CCI	64
MMBLKSIZ	Internal memory block size	64K
MEMRETRY	Number of retry after lock	0
MSGID	Message-id for messages	IDMS000
SYSID	Local SYSID for CCI	HOSTCCI
CCITRACE	Trace internal CCI calls	OFF

11.3.1.1 Input Parameter Descriptions

CCIMPNAM= CCI-memory-pool-name

CCIMPNAM

This parameter allows the user to overwrite the default name for the CCI memory pool.

CCI-memory-pool-name

This parameter represents a valid memory pool name of 8 characters maximum. It is useful only when two different releases of CCI would must coexist on the same machine.

Do not specify this parameter unless requested to do so by Computer Associates Technical Support.

CCIMPADR= CCI-memory-pool-address

CCIMPADR=

This parameter allows the user to force the CCI memory pool to start at a specific address. This parameter defaults to the lowest user address above the 16 MB line.

CCI-memory-pool-address

The CCI memory pool address must be the same for all the CCI users. The CCI memory pool address and the CCI memory pool size (see the **CCIMPSIZ** parameter) are only used by the first task, which enables the CCI memory pool: it can be the CAICCI task itself, or a CA-IDMS system. The other tasks connecting to the CCI memory pool use the same address.

If specified, this parameter has to be entered as an 8 hexadecimal digit string representing a valid address for a memory pool.

CCIMPSIZ= CCI-memory-pool-size**CCIMPSIZ**

This parameter allows the user to define the size of the CCI memory pool. The space within the CCI memory pool is managed by an internal memory manager. The space is physically allocated when needed only.

CCI-memory-pool-size

The value for this parameter may be specified in K- or M-bytes. Enter the number you want, followed by the letter K or M. The default value is equal to 16 Megabytes.

CCI#TSN= Maximum-number-of-TSN's**CCI#TSN**

This parameter allows the user to specify the maximum number of different TSN's that use the CCI services. This is needed because CCI maintains different tables for which the size is determined by the maximum number of TSN's which can be connected to the CCI memory pool.

Maximum-number-of-TSN's

A CCITCP driver processing different PC's counts for 1 TSN. A CA-IDMS system counts for 1 TSN (if running in mono-tasking mode), or for 2 or more TSN's (if running in multi-tasking mode).

MMBLKSIZ= Memory-block-size**MMBLKSIZ**

This parameter allows the user to define the standard memory block size that is used by the CCI internal memory manager. Only standard memory blocks are physically allocated and released.

Memory-block-size

The default value has been fixed to 64K and should satisfy all installations. The value for this parameter may be specified in K- or M-bytes. Enter the number you want, followed by the letter K or M.

MEMRETRY= Number-of-retry**MEMRETRY**

This parameter allows the user to define a retry counter that is used when the CCI memory manager tries to allocate a chunk of storage from a standard memory block which is locked at that time by another task.

Number-of-retry

When the retry counter reached the value 0, the next standard memory block is processed, and allocated if needed. The default value is 0.

MSGID= Message-id

MSGID=

This parameter allows the user to define the BS2000/OSD message identifier that is used for all the messages directed to SYSOUT or to the operator console.

Message-id

The message-id number (7 characters); by default, this parameter is equal to IDMS000.

SYSID= Local-sysid

SYSID=

This parameter represents the unique SYSID of the local host.

Local-sysid

The default value is HOSTCCI. It should not exceed 8 characters.

Presently, this parameter is not useful for BS2000/OSD because different hosts cannot be connected through CCI yet.

CCITRACE= CCI-trace-option

CCITRACE

This parameter allows the user to activate an internal trace of all the CCI calls which are processed for all CCI users.

CCI-trace-option

The possible values for this parameter are ON and OFF. The default is OFF.

Do not specify this parameter unless requested to do so by Computer Associates Technical Support.

11.3.2 CAICCI Control Commands

Once the CAICCI task is started, commands can be entered to control the way the CAICCI task is running, or to display information. Output from these commands is placed in the the SYSOUT system file assigned to the CAICCI task.

Commands may be entered using one of these methods:

- Via the INFORM-PROGRAM command — An INFORM-PROGRAM command must be entered from the console if the task is running in batch mode, or from the user terminal if running interactively.
- Via a job variable — To execute a command via a job variable:
 - Set the corresponding job variable value with the command you want to submit, starting in position 1.

- Only one blank character should separate each word or keyword of the command.
- Do not use any abbreviations other than those mentioned explicitly.

Job Variable Usage: During the startup of the CAICCI task, the first three positions of the job variable, if declared in the JCL, are cleared to blanks. At runtime, when a command is entered via that job variable, the value ACK or NAK is returned in position 1 through 3 (the other positions are not affected), to indicate if the command was executed successfully.

Job Variable Example: The startup JCL should contain the following command:

```
/SET-JV-LINK JV-NAME=JV.CAICCI, LINK-NAME=*JCAICCI
```

At runtime, from another terminal, we could enter the following command to shut down the CAICCI task:

```
/MODIFY-JV JC-CONTENTS=JV.CAICCI, SET-VALUE=C'CCI SHUT'
```

The output of this command is written to the SYSOUT system file of the CAICCI task. From another terminal, we can check if the command has been executed by entering the following command:

```
/SHOW-JV JC-CONTENTS=JV.CAICCI
```

The value returned should be %ACK SHUT, indicating that it has been executed successfully.

This table lists all the control commands and their abbreviations:

Console Command	Abbreviation
CCI HELP	N/A
CCI CCITRACE ON	N/A
CCI CCITRACE OFF	N/A
CCI RTIMER <u>time-value</u>	N/A
CCI CHECK	N/A
CCI CMD <u>BS2000-JCL-command</u>	N/A
CCI DISPLAY STORAGE	CCI D S

Console Command	Abbreviation
CCI DISPLAY ALL USERS	CCI D A U
CCI SHUTDOWN	CCI SHUT

11.3.2.1 CAICCI Command Descriptions

CCI HELP: This command displays the list of all the console commands which are accepted by the CAICCI task.

CCI CCITRACE ON / OFF: This command allows the user to set or reset the CCI trace option dynamically. Please see the CCITRACE input parameter description for more information.

CCI RTIMER time-value: As described before, at regular intervals, the CAICCI task starts a cleanup function call to free all resources that are not in use by users. A message is sent to SYSOUT for each TSN that is now inactive and still has CCI resources allocated.

When the CAICCI task starts, this interval is set to 5 minutes. This command allows the user to change the interval. The interval period should be expressed in seconds.

CCI CHECK: This command allows the user to activate the cleanup function described above immediately.

CCI CMD BS2000-JCL-command: This command allows the user to enter any BS2000/OSD JCL command which is supported by the CMD interface. For example, this allows the user to reassign the SYSOUT system file to another file, in order to read the original file without shutting down the CCI task.

CCI DISPLAY STORAGE: This command allows the user to display statistics on the CCI memory pool usage, including the percentage of its storage that is physically allocated.

CCI DISPLAY ALL USERS: This command displays all CCI users that are currently connected to CCI to the console and to SYSOUT. This refers to users that have outstanding receive requests on CCI.

The output of this command contains the sysid and application-id of the CCI user, the number of SEND and RECEIVE requests issued by that user, the corresponding TSN, and status flags.

CCI SHUTDOWN: This command terminates the CAICCI task, setting the CCI memory pool in an INACTIVE state. From that time, all CCI requests processed in the system are rejected.

11.4 CCITCP Task

The CCITCP task is a driver task that processes all the CCI requests coming from different PC's, i.e. it receives the CCI request, executes it on the mainframe, and sends the response back to the corresponding PC. The communication between each PC and the CCITCP driver is done using the socket program interface.

Different CCITCP drivers may be started on the same host. Each driver is assigned a specific TCP/IP port number, and processes the requests from all the PC's which have defined the same port number in their "Advanced TCP/IP Configuration" screen in the CCI configurator.

The CCITCP task is a driver task. It is switched to TP mode during its startup and should run with the same priority as the CA-IDMS system.

The CCI.LIB library contains the E-CCITCP JCL member that is used to start the CCITCP driver task.

To allow commands to be entered through a job variable, a user job variable must be defined in the startup JCL of the CCITCP task. The linkname associated with this job variable must be *JCCITCP.

```

/.CCITCP SET-LOGON-PARAMETERS
/MODIFY-JOB-OPTION LOGGING=(INFORMATION-LEVEL=MEDIUM)
/MODIFY-TEST-OPTIONS DUMP=YES
/SET-JV-LINK JV-NAME=jv.name2,LINK-NAME=*JCCITCP 1
/ASSIGN-SYSOUT TO=L.CCITCP.OUT
/ADD-FILE-LINK LINK-NAME=SERVICE,F-NAME=CCI.SERVICE.FILE
/ASSIGN-SYSDTA TO=*SYSCMD
/START-PROG FROM-FILE=*MOD(ELEM=CCITCP,LIB=CCI.LIB,RUN-MODE=*ADV)
enter your input parameters
END
/LOGOFF NOSPOOL

```

¹ This command is optional.

SERVICE File: The ADD-FILE-LINK card with the linkname SERVICE is optional. It allows the user to specify the TCP/IP port number via a service file.

The other way to specify the TCP/IP port number is via the TCPPOINT# input parameter (see later in this section). The SERVICE file is not checked if this parameter is specified.

The service file must be a SAM file, containing a record with the following structure: caicci port number/tcp, where port number represents the TCP/IP port number assigned to this driver. Please see the SERVICE source member in the CCI.LIB library for an example of port number specification.

11.4.1 CCITCP Input Parameters

The CCITCP program accepts input cards from SYSDTA with these limitations and characteristics:

- Only one parameter per input card is allowed.
- All input cards are free format.
- The input cards are composed of the parameter keyword followed by an equal sign and the parameter value.

This table lists all the valid parameters and their default values:

Keyword	Description	Default
CCIMPNAM	CCI memory pool name	CCIMP010
TCPPOINT#	TCP/IP port number	1202
TCP#SOCK	Maximum number of sockets	64
MSGID	Message-id for messages	IDMS000
CCITRACE	Trace CCI calls	OFF
DRVTRACE	Trace CCITCP driver	OFF
SOCTRACE	Trace sockets calls	OFF
MULTICON	Multiple connections allowed	NO

11.4.1.1 CCITCP Input Parameter Descriptions

CCIMPNAM= CCI-memory-pool-name

CCIMPNAM

This parameter allows the user to overwrite the default name for the CCI memory pool.

CCI-memory-pool-name

This parameter represents a valid memory pool name of 8 characters maximum. It is useful only in case two different releases of CCI would must coexist on the same machine.

Do not specify this parameter unless requested to do so by Computer Associates Technical Support.

TCPPOINT#=TCP/ IP-port-number

TCPPOINT#

This parameters allows the user to specify the TCP/IP port number assigned to this driver. This parameter is not needed is the TCP/IP port number is given via the SERVICE file.

IP-port-number

The default port number is 1202 for all the Computer Associates products.

TCP#SOCK= Number-of-sockets**TCP#SOCK**

This parameter allows the user to specify the maximum number of sockets this CCITCP driver is able to process.

Number-of-sockets

This is a way to limit the number of PC's which can connect to this CCITCP driver. The default value is 64.

MSGID= Message-id**MSGID=**

This parameter allows the user to define the BS2000/OSD message identifier that is used for all the messages directed to SYSOUT or to the operator console.

Message-id

The message-id number (7 characters); by default, this parameter is equal to IDMS000.

CCITRACE= CCI-trace-option**CCITRACE**

This parameter allows the user to activate an internal trace of all the CCI calls which are processed for all CCI users.

CCI-trace-option

The possible values for this parameter are ON and OFF. The default is OFF.

Do not specify this parameter unless requested to do so by Computer Associates Technical Support.

DRVTRACE= Driver-trace-option**DRVTRACE**

This parameter allows the user to activate an internal trace of the CCITCP driver task, i.e. all called functions, packets that are received and sent to the PC's, etc.

The possible values for this parameter are ON and OFF. The default is OFF.

Do not specify this parameter unless requested to do so by Computer Associates Technical Support.

SOCTRACE= Socket-trace-option**SOCTRACE**

This parameter allows the user to activate the BS2000/OSD socket trace.

Socket-trace-option

The possible values for this parameter are ON and OFF. The default is OFF.

MULTICON= Multiple-connection-option**MULTICON**

This parameter allows the user to open multiple connections from the same PC. When this option is not enabled, a second connection request forces the first connection to be closed.

Multiple-connection-option

The possible values for this parameter are YES and NO. The default is NO. This parameter replaces the old PCSIMUL parameter.

11.4.2 CCITCP Control Commands

Once the CCITCP task is started, commands may be entered to control the way the CCITCP task is running, or to display information. Output of these commands is always the SYSOUT system file from the CCITCP task.

Commands may be entered using one of these methods:

- Via the INFORM-PROGRAM command — An INFORM-PROGRAM command must be entered from the console if the task is running in batch mode, or from the user terminal if running interactively.
- Via a job variable — To execute a command via a job variable:
 - Set the corresponding job variable value with the command you want to submit, starting in position 1.
 - Only one blank character should separate each word or keyword of the command.
 - Do not use any abbreviations other than those mentioned explicitly.

Job Variable Usage: During the startup of the CCITCP task, the 3 first positions of the job variable, if declared in the JCL, are cleared to blanks. At runtime, once a command has been entered via that job variable, the value ACK or NAK is returned in position 1 through 3 (the other positions are not affected), to indicate the command has been executed successfully or not.

Job Variable Example: The startup JCL should contain the following command:

```
/SET-JV-LINK JV-NAME=JV.CCITCP, LINK-NAME=*JCCITCP
```

At runtime, from another terminal, we could enter the following command to shut down the CCITCP task:

```
/MODIFY-JV JC-CONTENTS=JV.CCITCP, SET-VALUE=C'CCITCP SHUT'
```

The output of this command is written to the SYSOUT system file of the CCITCP task. From another terminal, we can check if the command has been executed, by entering the following command:

```
/SHOW-JV JC-CONTENTS=JV.CCITCP
```

The value returned should be %ACKTCP SHUT, indicating that it has been executed successfully.

The following table lists all the control commands with their eventual abbreviation:

Console Command	Abbreviation
CCITCP HELP	N/A
CCITCP CCITRACE ON	N/A
CCITCP CCITRACE OFF	N/A
CCITCP DRVTRACE ON	N/A
CCITCP DRVTRACE OFF	N/A
CCITCP SOCTRACE ON	N/A
CCITCP SOCTRACE OFF	N/A
CCITCP CMD <u>BS2000-JCL-command</u>	N/A
CCITCP DISPLAY ALL USERS	CCITCP D A U
CCITCP DISPLAY ALL SOCKETS	CCITCP D A S
CCITCP SHUTDOWN	CCITCP SHUT

11.4.2.1 CCITCP Command Description

CCITCP HELP: This command displays the list of all the console commands that are accepted by the CCITCP task.

CCITCP CCITRACE ON / OFF: This command allows the user to set or reset the CCI trace option dynamically. Please see the CCITRACE input parameter description for more information.

CCITCP DRVTRACE ON / OFF: This command allows the user to set or reset the driver trace option dynamically. Please see the DRVTRACE input parameter description for more information.

CCITCP SOCTRACE ON / OFF: This command allows the user to set or reset the socket trace option dynamically. Please see the SOCTRACE input parameter description for more information.

CCITCP CMD BS2000-JCL-command: This command allows the user to enter any BS2000/OSD JCL command, which is supported by the CMD interface. For example, this would allow the user to reassign the SYSOUT system file to another file, in order to be able to read the old one without having to shut down the driver.

CCITCP DISPLAY ALL USERS: This command displays to the console and to SYSOUT all users that are currently connected to CCI via this driver.

The output of this command contains the port number and IP address of the corresponding PC, the sysid and the application-id of the CCI user, and the number of SEND and RECEIVE requests (expressed in number of bytes and in number of packets).

CCITCP DISPLAY ALL SOCKETS: This command displays to the console and to SYSOUT all sockets that are currently active for that driver.

The output of this command contains the socket number, the port number and IP address of the corresponding PC, the sysid and the application-id of the CCI user, and a status flag.

CCITCP SHUTDOWN: This command terminates the CCITCP task, closing all sockets connected to the different PC's.

11.5 IDMS Task

Three BS2KSTAR parameters allow a CA-IDMS system to be started before the CCI environment has been really initialized (CAICCI task), but still allowing that system to use the CCI services later without having to be recycled.

11.5.1 BS2KSTAR Input Parameters

The following table lists the new BS2KSTAR parameters with their default values:

Keyword	Description	Default
CCIMPNAM	CCI memory pool name	CCIMP010
CCIMPADR	CCI memory pool address	Lowest XS address
CCIMPSIZ	CCI memory pool size	16M

The meaning of these three new parameters is the same as the CAICCI task. Please refer to the CAICCI task description for an explanation of these parameters.

Special Notes: These parameters are not needed when the CAICCI task is started first. But if they are specified, they must match those parameters used to start the CAICCI task.

The CAICCI task must be active when the CCI line is opened in the IDMS system. The CCI line can be opened manually with the DCMT VARY LINE ON command, or automatically during startup if it is sysgenned as ENABLED. If this is not the case, the CCI line driver will abend and put a message into the log asking the user to check if the CCI environment is correctly started.

For more information on how to generate the CCI line, please refer to the *CA-IDMS System Generation* manual.

In case the CAICCI task has to be restarted (after an abend or normal shutdown), remember to close the CCI line first. This releases all the space allocated in the CCI memory pool.

Appendix A. BS2000/OSD Specific Modules

- A.1 System Modules Specific for BS2000/OSD A-3
- A.2 CA-Culprit Modules Specific for BS2000/OSD A-5

A.1 System Modules Specific for BS2000/OSD

ADSOBWTO	Extended batch: write to operator
ADSOLARC	Extended batch: log archiving
BS2KCMDL	STGID allowed BS2000 commands
BS2KDEVT	Device table
BS2KDKEY	PF-KEYS load utility
BS2KENTR	Entry-point for dynamically loaded modules
BS2KFETB	UTM front-end table
BS2KGPSI	Generalized SPOOL-INTERFACE utility
BS2KGSSM	Generate and prepare subsystem
BS2KOBJM	Object conversion utility
BS2KPTCH	Patch area module
BS2KSEM6	Presentation module for SEM62
BS2KSTAR	Central version startup module
BS2KSTID	STGID task processing
BS2KUDAS	UDAS program
BS2KUTMF	UCF-UTM: user format exit
BS2KUTMS	UCF-UTM: shutdown exit
IDMS	Primary interface for batch programs
IDMSBSVC	SVC-simulation
IDMSCANC	Abort routine for batch programs
IDMSDATE	TIME and DATE processing
IDMSDCB	Operating system dependent routines
IDMSIDMS	Secondary interface processing IDMS calls
IDMSINTB	Batch to CV old 10.2 interface
IDMSLBS2	Loader + 5C-trapping processing
IDMSOBS2	Operating system dependent routines (non-CV environment)
IDMSPBS2	Primary interface for batch programs
IDMSS120	DSSM module for storage protection on BS2000/OSD V3.0
IDMSTCM	UTM to CV interface

IDMSULIO	Utility I/O module
IDMS10XX	Main module for new secondary interface
RHDCBFMD	Extended batch: dump format
RHDCBIS	Extended batch: input processing
RHDCBMVT	Extended batch: vector table
RHDCBOS	Extended batch: output processing
RHDCCNOS	Change number of session
RHDCCOBI	COBOL support for DC-programs
RHDC00A	Common close routine for line-drivers
RHDCD00B	Line-driver for SYSINOUT terminals
RHDCD0LV	Line-driver for CCI
RHDCD03Q	Line-driver for OS 3270 simulation
RHDCD05D	Line-driver for DCAM terminals
RHDCD07Q	Line-driver for OS SYSOUT terminals
RHDCFSTB	UCF front-end table
RHDCMTLU	DCMT D/V LU processor
RHDCOTRC	Terminal trace manager
RHDCO00B	Open routine for SYSINOUT terminals
RHDCO03Q	Open routine for OS 3270 simulation
RHDCO05D	Open routine for DCAM terminals
RHDCO07Q	Open routine for OS SYSOUT terminals
RHDCPCBO	COBOL program processor selection IBM/SIEMENS prog.
RHDCUBAT	UCF-BATCH : PRINT=NO
RHDCUCFB	UCF-BATCH : PRINT=YES
RHDCUCFT	UCF-TIAM
RHDCUCFU	UCF-UTM
WTOEXIT	Exit routine for console messages
WTOREXIT	Exit routine for console messages with reply

A.2 CA-Culprit Modules Specific for BS2000/OSD

CULE
CULEMLIN
CULL
CULLCOMP
CULLMTCH
CULLPGEN
CULLUS00
CULLUS01
CULLUS10
CULLUS22
CULPPDSR
CULP0

Appendix B. Error Messages and Abend Codes

- B.1 Error Messages from the DCAM Driver B-3
- B.2 Error Messages from SEM62 B-5
- B.3 Return Codes from BS2KDSCV B-6
- B.4 Return Codes from the Library Interface B-7
- B.5 Abend Codes from BS2KSRVD B-9
- B.6 Abend Codes B-10

B.1 Error Messages from the DCAM Driver

DC085100 LINE <line-id> SHORTAGE ON STORAGE FOR LOGON AREA
CAUSES REJECTION OF LOGON REQUEST

DC085101 LINE <line-id> INQUIRE ERROR <error code> DURING LOGON.

DC085102 LINE <line-id> NO FREE PTE FOUND TO LOG ON <terminal-id>

DC085103 LINE <line-id> TRANSDATA DEVICE TYPE <device-type-name>
NOT SUPPORTED FOR TERMINAL <terminal-id>.

DC085104 LINE <line-id> OPEN CONNECTION ERROR <error-code>
DURING LOGON OF TERMINAL <terminal-id>.

DC085105 LINE <line-id> YREJLOG MACRO FAILED WITH ERROR
<error-code> FOR TERMINAL <terminal-id>.

DC085108 LINE <line-id> PRINTER <terminal-id> CANNOT BE ACQUIRED.
REQUEST DELAYED.

DC085109 DCAM LINE <line-id> READY TO OPERATE WITH
APPLICATION ID <appl-id>.

DC085110 LINE <line-id> TERMINAL <terminal-id> HAS BEEN LOGGED ON

DC085111 LINE <line-id> OPCON ERROR <error-code> DURING ACQUIRE
OF TERMINAL <terminal-id>.

DC085113 LINE <line-id> PTE-DEVICE TYPE <device-type> NOT
SUPPORTED FOR TERMINAL <terminal-id>.

DC085120 LINE <line-id> TERMINAL <terminal-id> HAS BEEN ACQUIRED.

DC085121 LINE <line-id> CLSCON ERROR <error-code> DURING
VARY-OUT OF TERMINAL <terminal-id>.

DC085130 LINE <line-id> TERMINAL <terminal-id> HAS BEEN LOGGED
OFF

DC085131 LINE <line-id> CONNECTION OF TERMINAL <terminal-id> HAS
BEEN CLOSED BY THE USER.

DC085132 LINE <line-id> CONNECTION OF TERMINAL <terminal-id> HAS
BEEN CLOSED BY DCAM, REASON <reason-code>.

DC085139 LINE <line-id> DCAM APPLICATION <application-name> HAS
BEEN CLOSED, REASON <reason-code>.

DC085140 LINE <line-id> DCAM-APPLICATION <application-name> HAS
BEEN CLOSED BY THE NETWORK ADMINISTRATOR OR THE
SYSTEM OPERATOR.

DC085141 LINE <line-id> SEND ERROR <error-code> to TERMINAL
<terminal-id>.

DC085142 LINE <line-id> DATASTREAM CONVERSION ERROR
<error-code> FOR TERMINAL <terminal-id>.

DC085143 LINE <line-id> NACK RECEIVED FROM TERMINAL
<terminal-id>

DC085144 LINE <line-id> CHECKSEND ERROR, FEEDBACK
<feedback-string> FOR TERMINAL <terminal-id>.

DC085145 LINE <line-id> CHECKSEND ERROR; FEEDBACK
<FEEDBACK-string> FOR TERMINAL <terminal-id>.

DC085146 LINE <line-id> AN ACKNOWLEDGMENT HAS BEEN RECEIVED
FOR A CONNECTION THAT IS NOT KNOWN TO THE DCAM
DRIVER.

DC085147 LINE <line-id> INTERVENTION REQUIRED ON <device-id>.
DC085148 LINE <line-id> WRITE REQUESTED AND NO WRITE BUFFER IN
PTE FOR TERMINAL <terminal-id>.
DC085150 LINE <line-id> DCAM DRIVER OPEN FUNCTION <function>
FAILED WITH ERROR CODE = <error-code>.
DC085161 LINE <line-id> YSEND (FOR READ WITHOUT WAIT FOR ATTN)
ERROR <error-code> FOR TERMINAL <terminal-id>.
DC085162 LINE <line-id> A DCAM MESSAGE HAS BEEN RECEIVED
WITHOUT THE NECESSARY USER FIELD INFORMATION.
DC085164 LINE <line-id> YSEND (FOR PF-KEY SIMULATION) ERROR
<error-code> FOR TERMINAL <terminal-id>.
DC085166 LINE <line-id> DATA STREAM CONVERSION ERROR
<error-code> FOR TERMINAL <terminal-id>.
DC085167 LINE <line-id> YRECEIVE DCAM MACRO FAILED WITH
FEEDBACK = <feedback>.
DC085168 LINE <line-id> YRECEIVE DCAM MACRO RETURNED AN
UNEXPECTED FEEDBACK = <feedback> FOR TERMINAL
<terminal-id>.
DC085169 LINE <line-id> RPB SHORTAGE DURING SETUP OF AN
YSEND(SM) MACRO FOR TERMINAL <terminal-id>.
DC085170 LINE <line-id> STORAGE SHORTAGE (#GETSTG) TO BUILD AN
INPUT BUFFER FOR TERMINAL <terminal-id>.
DC085180 YINQUIRE DCAM MACRO FAILED IN LOGON CONTINGENCY
ROUTINE.
DC085181 YREJLOG DCAM MACRO FAILED IN LOGON CONTINGENCY
ROUTINE
DC085182 LOGON REJECTED : CV QUIESCED OR DCAM LINE NOT
READY
DC085183 NO PLE FOUND WITH SAME AID DURING DCAM LOGON
DC085184 MISMATCHING DETECTED IN LOSCON CONTINGENCY
ROUTINE
DC085185 SEQUENCE NUMBER OF TACK NOT FOUND IN ANY RMODE
DC085186 MISMATCH DETECTED IN DCAM TRANSPORT
ACKNOWLEDGMENT CONTINGENCY ROUTINE>
DC085187 USERFLD FROM CCB MISSING IN DCAM TACK
CONTINGENCY
DC085188 CONNECTION ID IS MISSING IN DCAM TACK CONTINGENCY
DC085189 DCAM COMEND ACTIVATED AND NO LINE TO CLOSE
DC085190 GO SIGNAL CONTINGENCY WITHOUT CORRESPONDING
FDBK
DC085191 GO CONTINGENCY FOUND MISMATCH IN PTE/RNODE LINKS
DC085192 PROCON REJECTED ; CV QUIESCED OR DCAM LINE NOT
READY
DC085193 NO PLE FOUND WITH SAME AID DURING DCAM PROCON
DC085194 RPB SHORTAGE IN DCAM LOGON OR PROCON
CONTINGENCY

B.2 Error Messages from SEM62

- DC084420 UNABLE TO ALLOCATE A DCAM PTE SATISFYING THE REQUESTOR.
- DC084421 UNABLE TO ALLOCATE A D0FI PTE SATISFYING THE REQUESTOR.
- DC086020 CNOS ERROR: INVALID REQUEST.
- DC282021 INVALID DCMT VARY LU MAXSES COMMAND.
- DC282022 INVALID DCMT VARY LU SESSION COMMAND.
- DC282023 INVALID DCMT VARY LU COMMAND.
- DC282024 LOGICAL UNIT <luname> <primary/secondary> NOT FOUND.
- DC282025 VARY LU COMMAND ABORTED. CNOS CONVERSATION FAILED.
- DC282026 MAXSES EXCEEDS NUMBER SYSGENED PTERMS IN SOURCE SYSTEM.
- DC282027 MAXSES EXCEEDS NUMBER SYSGENED PTERMS IN REMOTE SYSTEM.
- DC282028 REQUESTED NUMBER OF SESSIONS EXCEEDS CURRENT MAXSES VALUE.
- DC282029 MAXSES VALUE NEGOTIATED TO <number> BY SOURCE LOGICAL UNIT.
- DC282030 MAXSES VALUE NEGOTIATED TO <number> BY REMOTE LOGICAL UNIT.
- DC282031 VARY LU INITIALIZE COMMAND ABORTED. RESET LOGICAL UNIT FIRST.

B.3 Return Codes from BS2KDSCV

BS01

Explanation: Invalid parameters

BS02

Explanation: Command code of 3270-string in error

BS03

Explanation: Length of input string > 4096

BS04

Explanation: Output string would exceed output buffer

BS05

Explanation: More than 10 attributes pairs

B.4 Return Codes from the Library Interface

E002

Explanation: End of file on input

E003

Explanation: Invalid function

E004

Explanation: LMS error

E005

Explanation: BS2KVCOB error

E006

Explanation: Invalid open type

E007

Explanation: Maximum number of files reached

E008

Explanation: Member already opened

E009

Explanation: Close error

E010

Explanation: Member not found

E011

Explanation: GETN with library type=C

E012

Explanation: Open error on temporary file

E013

Explanation: Invalid "prefix,filename"

E014

Explanation: Cannot create temporary file

E015

Explanation: File already closed

E016

Explanation: PUTN with library type=I

E017

Explanation: Output file locked

E018

Explanation: Invalid library type, or no LMS library

E019

Explanation: Link to LMSUP in LMSLIB failed

W020

Explanation: LMS warning : concatenated object written to SRCLIB

W021

Explanation: LMS warning : non-object module written to SRCLIB

W022

Explanation: LMS warning : member name truncated to 8 characters

B.5 Abend Codes from BS2KSRVD

- BS94** A request to a BS2000/OSD service was done with an invalid parameter list. Usually this means storage was overwritten.
- BS95** A request to a BS2000/OSD service was done and the service is not available because it abended or it can not be restarted. The administrator has to correct the cause of the abend. Once corrected, the administrator can use the SRVADMIN administrator task to restart the service.
- BS96** The service module for a BS2000/OSD service either:
- Is linked incorrectly
 - Does not contain the requested entry point
- Eventually, correct the service definition and regenerate the service module.
- BS97** All ERE's are in use. Retry later or increase the number of ERE's.
- BS98** An eventing problem (ENAEI or POSSIG) problem occurred during startup of a BS2000/OSD service. Most likely causes are overwritten storage or maximum number of events reached.
- BS99** The /ENTER of a BS2000/OSD service failed. Probably the service definition parameters ENTER-FILE-NAME and/or ENTER-PARMS are incorrect. Eventually correct the service definition and regenerate the service module.

B.6 Abend Codes

4815

Explanation: Internal error in BSVC function 11.

Module: IDMSBSVC

4922

Explanation: ENAEI/ENACO/SOLSIG SVC has failed. A CV with the same CV number is already active or after an abend of this CV, some UCF terminals are still active.

Module: RHDCOS00

4928

Explanation: STCK has failed in the TIME routine.

Module: RHDCOS00

4929

Explanation: The post of a subtask or maintask by the postexit has failed.

Module: RHDCOS00

4930

Explanation: Enable of the internal communication has failed

Module: RHDCOS00

4931

Explanation: After the subtask is started, it posts the maintask. This post SVC from subtask to maintask has failed.

Module: RHDCOS00

4932

Explanation: Disable of the event used to inform the maintask that the subtask is up, has failed.

Module: RHDCOS00

4933

Explanation: The TSN dependent stack space has overflowed.

Module: BS2KSTAR

4934

Explanation: ABEND STXIT class has been activated (CANCEL, LOGOFF, EXEC).

Module: RHDCOS00

4935

Explanation: ENAMP of the XA pool processed by a subtask has failed.

Module: RHDCOS00

4936

Explanation: POSSIG failed in POST routine.

Module: RHDCOS00

4938

Explanation: Internal error in activation of contingencies.

Module: RHDCOS00

4939

Explanation: A central version with the same CVNUM is already active.

Module: RHDCOS00

Index
