Communications

Bull DPS 7000

Distributed Operator Facility - Programmed Operator User's Guide

---

Subject: This manual describes the programming interface for the Distributed Operator Facility - Programmed Operator (DOF 7-PO). It provides the information needed to write programmed operator applications for GCOS 7 using GPL primitives or C language functions.

Special Instructions: Revision 06 replaces Revision 05 for all users of GCOS 7-V9 (TS9764).

Software Supported: GCOS 7-V9 (TS9764).

Software/Hardware required:

Date: August 1999

Bull S.A.
CEDOC
Atelier de reprographie
34, rue du Nid de Pie BP 428
49004 ANGERS Cedex 01
FRANCE

Bull HN Information Systems Inc.
Publication Order Entry
FAX: (978) 294-7411
MA30/865A
Technology Park
Billerica, MA 01821
U.S.A.

47 A2 80UC Rev06

Your suggestions and criticisms concerning the form, contents and presentation of this manual are invited.
A form is provided at the end of this manual for this purpose.

47 A2 80UC Rev06

Preface

Scope and Objectives

This document describes the programming interface for the Distributed Operator Facility - Programmed Operator (DOF 7-PO). It provides the essential information needed to write programmed operator applications for GCOS 7 using GPL primitives or C language functions.

The reader should use this manual in conjunction with the other GCOS 7 manuals listed below (see Related Documents), since the information on GCOS 7 commands and messages contained in them is a prerequisite for this User's Guide.

Intended Readers

This manual is intended for the programmer who needs to write a programmed operator application. Familiarity with GPL or C, and knowledge of GCL commands (as contained in the *System Operator's Guide* and in *Network Operations*), is assumed.

Structure

Section 1	Gives an overview of DOF 7-PO.
Section 2	Explains how DOF 7-PO works from the point of view of the application.
Section 3	Explains how DOF 7-PO transfers data, and introduces some special terms.
Section 4	Gives examples of how to code the various steps of a DOF 7-PO application.
Section 5	Is a reference to the GPL primitives.
Section 6	Is a reference to the C functions.
Appendix A	Provides cross-reference tables to the components, commands, responses, and messages supported by DOF 7-PO.
Appendix B	Is a listing of an example application program in GPL and in C.

Appendix C	Is a listing of an example application program using filters.
Appendix D	Is a listing of an example application program that communicates with a remote system.

Bibliography

Structured Records (OMH Format) Part 1 - Commands *47 A2 81UC*
Structured Records (OMH Format) Part 2 - Messages *47 A2 82UC*
Structured Records (DSAC Format) *47 A2 83UC*
System Operator's Guide (V6) *47 A2 60UU*
System Operator's Guide (V7) *47 A2 47US*
System Operator's Guide (V8 and V9) *47 A2 53US*
Console Messages *47 A2 61UU*
System Overview *47 A2 04UG*
GPL System Primitives *47 A2 34UL*
ARM User's Guide *47 A2 11US*
Network Operations (V6) *47 A2 72UC*
Network User's Guide (V9) *47 A2 94UC*
DSAC User's Guide *47 A2 75UC*
C Language User's Guide *47 A2 60UL*
C Primitives *47 A2 64UL*
GPL User's Guide *47 A2 36UL*
TDS Administrator's Guide (V6) *47 A2 20UT*
TDS Administrator's Guide (V7, V8 and V9) *47 A2 32UT*

Syntax Notation

The following conventions are used for presenting programming syntax:

ITEM	An item in upper case is a literal value, to be specified as shown. The upper case is merely a convention; in practice you can specify the item in upper or lower case.
item	An item in lower case is a non-literal, indicating that a user-supplied value is expected. In most cases it gives the type and maximum length of the value: **char105** a string of up to 105 alphanumeric characters **name31** a name of up to 31 characters **lib78** a library name of up to 78 characters **file78** a file name of up to 78 characters **star31** a star name of up to 31 characters In some cases, it gives the format of the value: **a** a single alphabetic character **nnn** a 3-digit number **hh.mm** a time in hours and minutes

	In other cases, it is simply descriptive of the value: **device-class** **condition** **any-characters**
ITEM	An underlined item is a default value. It is the value assumed if none is specified.
bool	A boolean value which is either 1 or 0. A boolean parameter can be specified by its keyword alone, optionally prefixed by "N". Specifying the keyword alone always sets the value to 1. Prefixing the keyword with "N" always sets it to 0.
{item | item }	Braces indicate a choice of values. Only one can be selected.
[ ]	Square brackets indicate that the enclosed item is optional. An item not enclosed in square brackets is mandatory.
( )	Parentheses indicate that a single value or a list of values can be specified. A list of values must be enclosed by parentheses, with each value separated by a comma or a space.
...	Ellipses indicate that the item concerned can be specified more than once.
+ = $ * / - .	Literal characters to be specified as shown.

EXAMPLE 1:

```
[                                ]
[        { IMMED                 }  ]
[  WHEN= { [dd.mm.yy.]hh.mm      }  ]
[        { +nnnn{W D H M}        }  ]
[                                ]
❑
```

This means you can specify:

1. Nothing at all (WHEN=IMMED applies).
2. WHEN=IMMED (the same as nothing at all).
3. WHEN=22.30 to specify a time (today's date assumed).
4. WHEN=10.11.87.22.30 to specify a date and time.

5. WHEN=+0002W to specify 2 weeks from now.
6. WHEN=+0021D to specify 21 days from now.
7. WHEN=+005H to specify 5 hours from now.
8. WHEN=+0123M to specify 123 minutes from now.

EXAMPLE 2:

```
PAGES=(dec4[-dec4] ...)
```

❑

Indicates that PAGES must be specified. Valid entries are a single value, or a list of values enclosed in parentheses. The list can consist of single values separated by a comma (or space), a range of values separated by a hyphen, or a combination of both. For example:

PAGES=(2,4,10-25,33-36,78,83)

EXAMPLE 3:

```
[ REPLACE ={ bool | 0 } ]
```

❑

This is a boolean parameter whose default value is zero. You can specify:

1. Nothing at all (REPLACE=0 applies)
2. REPLACE=0 or simply NREPLACE
3. REPLACE=1 or simply REPLACE

Table of Contents

Table of Graphics

Figures

Tables

1. Introduction to DOF 7-PO

1.1 What is DOF 7-PO?

DOF 7-PO (Distributed Operator Facility - Programmed Operator) is a software product that makes possible the automation of operator tasks that would otherwise require the presence of a dedicated human operator.

The software provides a GPL or C language programming interface that allows a user application to send commands to be executed by a local or remote GCOS 7 system, and to receive messages in response.

A typical Programmed Operator application can manage all or part of a DPS 7000 system, or several systems, performing tasks such as:

- starting jobs for users
- reacting to errors
- managing the network
- monitoring system resources
- reporting network activity.

1.2 Why Use DOF 7-PO?

A DPS 7000 operator with MAIN access rights can use a set of privileged commands to monitor and control system and network operation. On large or multisystem sites, there may be several main operators, each responsible for a particular aspect of system behaviour, for example, network administration, batch operations, or responding to system events. Each operator will then have access to a specific set of commands relating to that task. Only those system messages associated with that task will be sent to that operator's console.

A Programmed Operator application can be written to automate the function of a main operator, (or an operator without MAIN access rights, as in the example in Appendix B.1). The application can then make decisions and take action based on the information it receives. For example, if the system is overloaded with too many jobs, it can hold some of them. If resources requested by a job are not available or unknown, it can cancel the job.

The aim of these types of applications is to reduce to a minimum the amount of human intervention required. A process executing on the DPS 7000 does not have to stop and wait for an aknowledgement, if the human operator is not there to reply. It is not necessary for a human operator to make a decision based on his or her own judgement, since the program can make the same decision automatically, based on the data available to it.

This should increase the reliability and availability of the system. It also frees the operator for other tasks.

A Programmed Operator application may be especially useful to control smaller DPS 7000 systems at sites where a dedicated operator is not available, or when the processing done by the system is repetitive and predictable enough to make the presence of a human operator unnecessary.

The DOF 7-PO programming interface can be used to create applications to manage OSI/DSA networks. In particular, a DOF 7-PO application can receive, in real time, all the events sent over the network (see the *DSAC User's Guide* for specific information).

2. How DOF 7-PO Works

NOTE:
Words or phrases in **bold** are terms which appear in the glossary at the end of the manual.

2.1 Commands, Responses, and Unsolicited Messages

A Programmed Operator application controls the DPS 7000 or the network by using the DOF 7-PO interface GPL primitives or C language functions to send **commands**, and to receive **responses**.

Also, the application may receive a certain number of **unsolicited messages** (also called unsolicited events in networking). These are messages that are not related to any command sent by the application. The application receives or does not receive these messages, depending on the **filters** it has defined.

The application is therefore a 'programmed operator' that functions in much the same way as a human operator. A Programmed Operator application is essentially "**event**-driven", reacting to messages according to their relative importance, and issuing commands to monitor the current situation and modify it if necessary.

All of the commands, responses, and unsolicited messages that are to be interpreted by the application must be declared in the program environment. This is explained in the section "How to Code a Programmed Operator Application".

2.2 Objects and Object Managers

DOF 7-PO uses the term **object** to refer to any system or network entity that can be the object of a command. For example, a job running for a user is an object. So is a session linking a terminal to an application, and so is the physical line or cable used by the session.

A Programmed Operator application sends a command to an **object manager**, which is the software component responsible for a particular object. The object manager processes the command, and sends the appropriate response back to the application.

The CJ (Cancel Job) command, for instance, is processed by an **object manager** called Job Management/Scheduling (JOBM/SCH). The JOBM/SCH object manager returns one of the following messages in response to the command:

SH04, SH09, SH11, SH13, SH26, SH45

The object manager is in fact transparent to the application. It is only visible in the sense that the message identifier begins with two characters that indicate which object manager produced the message (SH for scheduler in this example).

Appendix A of this Manual (Cross-reference Tables) contains a series of tables that show for each available command the possible responses to it. Appendix A also includes a listing of the layout and contents of each message.

2.3 DOF 7-PO from the Application Point of View

The following diagram shows a Programmed Operator application communicating with the local system, and with remote systems **by means of the DOF 7-PO interface**.

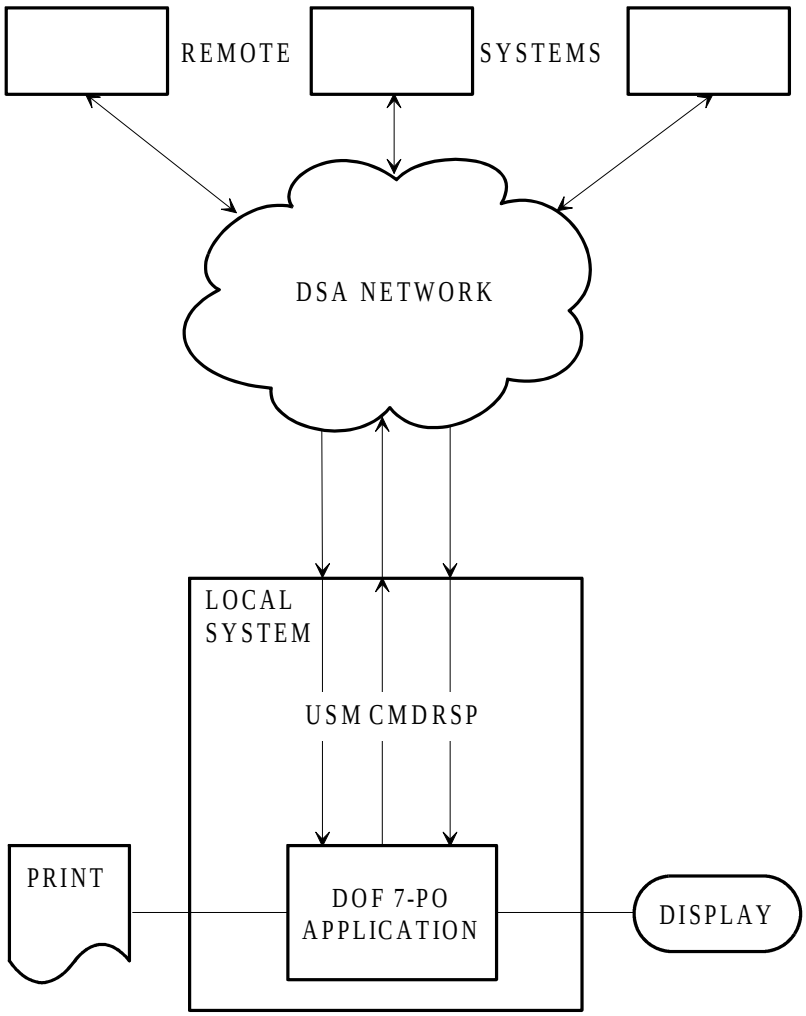

Figure 2-1. A Programmed Operator Application

3. How DOF 7-PO Transfers Data

This section introduces some more DOF 7-PO-specific terminology, and explains how DOF 7-PO transfers information between the application and the object manager.

3.1 GCOS 7 Structured Records

The basic means for the transfer of the contents of a command, a response, or an unsolicited message, is the **GCOS 7 structured record**.

There is one structured record for each command, response, and unsolicited message supported by DOF 7-PO. The formal definition of the records can be found in the *Structured Records* manuals.

DOF 7-PO transfers data in a transparent way; that is, it does not interpret the contents of the records sent, but simply ensures delivery and data integrity. The GCOS 7 structured record contains the name of the **format manager**, which is the GCOS 7 system software that interprets the contents of the records. It also contains the description of the data in the record.

This mechanism ensures that the records are self-descriptive and independent of the software that interprets them. The records do not need to be changed if another format manager is added in the future.

DOF 7-PO supports two format managers:

OMH	(Operator Message Handler), which is the GCOS 7 system format manager. OMH interprets the records exchanged between DOF 7-PO and object managers belonging to the GCOS 7 system, whether on the local system or on remote systems.
DSAC	(Distributed Systems Administration and Control), which is the network format manager. DSAC interprets the records exchanged between DOF 7-PO and network object managers.

3.2 DOF 7-PO Objects

DOF 7-PO is both a programmatic interface and a protocol. The basic unit of data transfer used by the DOF 7-PO protocol is the **DOF 7-PO object**.

The term DOF 7-PO object means a structured record containing a command, a response, or an unsolicited message.

The object always consists of two parts:

the **descriptor** which names the format manager and describes the data structure. This is the fixed part of the object.

the **record** is the variable part of the object. It contains the data to be transferred in one to three **regions**. The regions are:

Selection: which identifies the target of the command (commands only)
Modification: which contains the new attributes for the target of the command (commands only)
Response: which contains the parameters that make up the response to a command, or which contains an unsolicited message (response or unsolicited message only).

Each region begins with a **presence mask**, which indicates whether the fields in that region are significant or not. The presence mask is followed by the fields that make up the region.

The DOF 7-PO object therefore has this layout for a command:

Descriptor (fixed)	Presence Mask	Selection Region	Presence Mask	Modif. Region

or this layout, for a response or unsolicited message:

Descriptor (fixed)	Presence Mask	Response Region

There may be several **parameters**, or different combinations of elementary fields within the region (see the example of the CJ command below). In the case of a command, the fields correspond to the GCL parameters.

3.3 Example of Structured Record for CJ Command

The following example is taken from *Structured Records (OMH Format) Part 1 - Commands*. It shows the structure associated with the CJ DOF 7-PO command, which corresponds to the CANCEL_JOB GCL command.

The formal definition of the record, as it appears in *Structured Records (OMH Format) Part 1 - Commands*, is on the left of the page. An explanation of the various parts of the record is on the right of the page. See also the comments in paragraph *"Comments on the Example Record"* for further explanation.

```
-------------------------------------------------------
Structured Record                    Explanation
-------------------------------------------------------
 /* CJ
      { RONS=ron1[ron2][ron3][ron4][ron5][ron6][ron7][ron8]
        [ron9][ron10][ron11][ron12] [ron13][ron14][ron15][ron16]
        |   ALL SELECT= [jclass] / [usernm] / [projnm] }
    [ { STRONG   |   FORCE   |   ENDSTEP  } ]
    [ { dump } ]
    [ { MSSG=mssg } ]
    [ { JOBSTATE=jstate } ]
    [ { JOBNAME=jnm } ]  */
                                         Comment showing command
                                         options (parameters).
DCL 1 PMOS_CJ_REC,
      2 SEL,                             Selection region begins.
       3 PRESENCE_MASK,                  Presence mask begins.

        4 P1ALTER_ON LOGBIN(8),          Parameter 1 flag.
        4 RON1_ON LOGBIN(8),             Presence mask for
        4 RON2_ON LOGBIN(8),             parameter 1 begins.
        4 RON3_ON LOGBIN(8),
        4 RON4_ON LOGBIN(8),
        4 RON5_ON LOGBIN(8),
        4 RON6_ON LOGBIN(8),
        4 RON7_ON LOGBIN(8),
        4 RON8_ON LOGBIN(8),
        4 RON9_ON LOGBIN(8),
        4 RON10_ON LOGBIN(8),
        4 RON11_ON LOGBIN(8),
        4 RON12_ON LOGBIN(8),
        4 RON13_ON LOGBIN(8),
        4 RON14_ON LOGBIN(8),
        4 RON15_ON LOGBIN(8),
        4 RON16_ON LOGBIN(8),
        4 ALL_ON LOGBIN(8),
        4 JCLASS_ON LOGBIN(8),
        4 USERNM_ON LOGBIN(8),           Presence mask for
        4 PROJNM_ON LOGBIN(8),           Parameter 1 ends

        4 P2ALTER_ON LOGBIN(8),          Parameter 2 flag.
        4 STRONG_ON LOGBIN(8),           Presence mask.
        4 FORCE_ON LOGBIN(8),
        4 ENDSTEP_ON LOGBIN(8),
```

```
          4 P3ALTER_ON LOGBIN(8),          Parameter 3 flag.
          4 DUMP_ON LOGBIN(8),             Presence mask.

          4 P4ALTER_ON LOGBIN(8),          Parameter 4 flag.
          4 MSSG_ON LOGBIN(8),             Presence mask.

          4 P5ALTER_ON LOGBIN(8),          Parameter 5 flag.
          4 JSTATE_ON LOGBIN(8),           Presence mask.

          4 P6ALTER_ON LOGBIN(8),          Parameter 6 flag.
          4 JNM_ON LOGBIN(8),              Presence mask.
                                           Presence mask ends.

                                           Record fields begin
        3 P1ALTER LOGBIN   (  8),          Parameter 1
        3 RON1 CHAR     (  8),
        3 RON2 CHAR     (  8),
        3 RON3 CHAR     (  8),
        3 RON4 CHAR     (  8),
        3 RON5 CHAR     (  8),
        3 RON6 CHAR     (  8),
        3 RON7 CHAR     (  8),
        3 RON8 CHAR     (  8),
        3 RON9 CHAR     (  8),
        3 RON10 CHAR     (  8),
        3 RON11 CHAR     (  8),
        3 RON12 CHAR     (  8),
        3 RON13 CHAR     (  8),
        3 RON14 CHAR     (  8),
        3 RON15 CHAR     (  8),
        3 RON16 CHAR     (  8),
        3 ALL CHAR     (  1),
        3 JCLASS CHAR     (  2),
        3 USERNM CHAR     ( 12),
        3 PROJNM CHAR     ( 12),

        3 P2ALTER LOGBIN   (  8),          Parameter 2
        3 STRONG CHAR     (  1),
        3 FORCE CHAR     (  1),
        3 ENDSTEP CHAR     (  1),

        3 P3ALTER LOGBIN   (  8),          Parameter 3
        3 DUMP LOGBIN   (  8),

        3 P4ALTER LOGBIN   (  8),          Parameter 4
        3 MSSG CHAR     (255),

        3 P5ALTER LOGBIN   (  8),          Parameter 5
        3 JSTATE LOGBIN    (  8),

        3 P6ALTER LOGBIN   (  8),          Parameter 6
        3 JNM CHAR         (  8);

      ------------------------------------------------------
```

3.3.1 Comments on the Example Record

It is useful to compare this structured record with the description of the CANCEL_JOB command in the *System Operator's Guide*.

The DOF 7-PO CJ command record contains only the selection (SEL) region, since the purpose of the command is simply to cancel certain selected jobs, with various parameters provided as options. Other commands may have a modification region to contain the new values to be sent.

The PnALTER_ON field indicates whether the parameter is active or not. In this example, the first parameter allows the program to specify the RONs to cancel, either by naming up to 16 RONs individually, or by using the SELECT option to specify a particular job class, user, or project.

Parameter 2 specifies when the command takes effect. Parameter 3 controls whether a memory dump is produced. Parameter 4 sends a message concerning the cancelled job. Parameter 5 specifies the jobs to be cancelled, by their current state. Parameter 6 specifies the jobs to be cancelled by name.

More than one parameter may be active at the same time.

3.3.2 How to Initialize a Record

When you initialize a structured record in order to send a command, the procedure to follow is:

- set each byte of the presence mask related to a significant field of that region to "01"X, and reset the others to "00"X.
- give a value to each of the fields, taking care not to specify any mutually-exclusive combinations (such as FORCE and ENDSTEP in CANCEL_JOB).

You do not need to set the PnALTER_ON and PnALTER fields when you send a command. The format manager sets them automatically to the right value, assuming that the command is accepted.

4. How to Code a Programmed Operator Application

4.1 Flowcharts

Before examining the steps involved in writing a Programmed Operator application, it is useful to know something about the mechanism used by DOF 7-PO for queuing commands, responses, and unsolicited messages.

4.1.1 Chains of Commands and Related Responses

The following flowchart shows the basic principle of sending a command and receiving the responses to it.

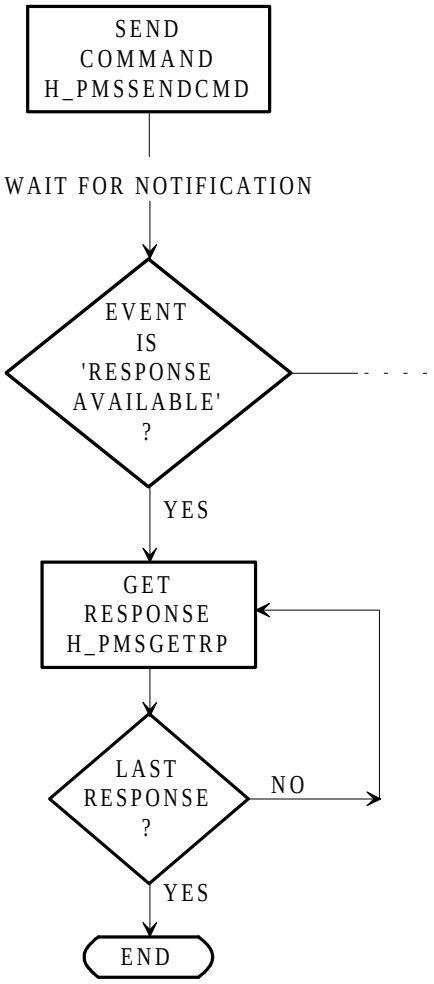

Figure 4-1. Sending a Command/Receiving Responses

A Programmed Operator application submits a command in structured format using the H_PMSSENDCMD (Send Command) GPL primitive or C function. The command is stored in the **DOF 7-PO queue file**, while DOF 7-PO calls the appropriate object manager. The application then waits for notification of the event indicating that the response is available.

The object manager processes the command, and generates the responses to it. These are also stored in the queue file.

After the last response by the object manager, the application is notified, and the **state** of the chain of a command and its related responses becomes 'to be delivered'. This means that the whole set of responses has been linked to the command, and is ready to be received by the application. The application can then obtain the responses one at a time with the H_PMSGETRP (Get Response) primitive or function (see the following diagram).

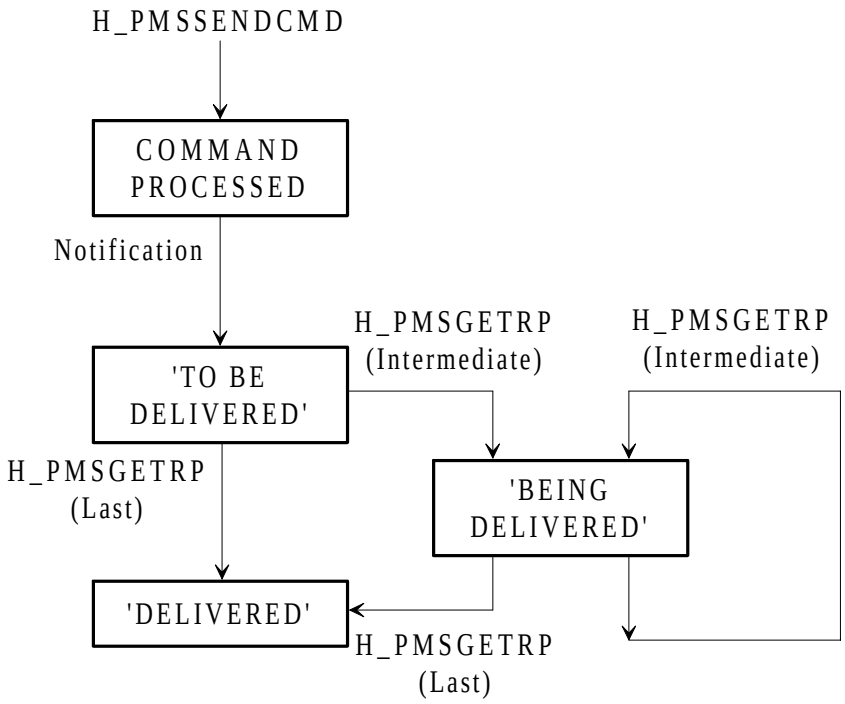

Figure 4-2. State of a Chain of Command/Responses

Once the state of the chain becomes 'delivered', the whole chain of command and related responses is then deleted from the queue file. Until this happens, it is always possible for the application to use the H_PMSGETRPFR GPL primitive or C function to get the name of the record related to the first response again, and thereby fetch all the responses again from the beginning. This can be useful if an error occurs while fetching the responses.

As the first response is delivered, the state of the chain becomes 'being delivered'. As the last response is delivered, the state becomes 'delivered', indicating that all responses have been received. The application can test for these last two states with the STATE parameter of H_PMSGETRP.

The whole chain of command and related responses is then deleted from the queue file.

4.1.2 Chains of Unsolicited Messages

This diagram shows the principle of receiving unsolicited messages. The application waits for notification of the event indicating that an unsolicited message is available.

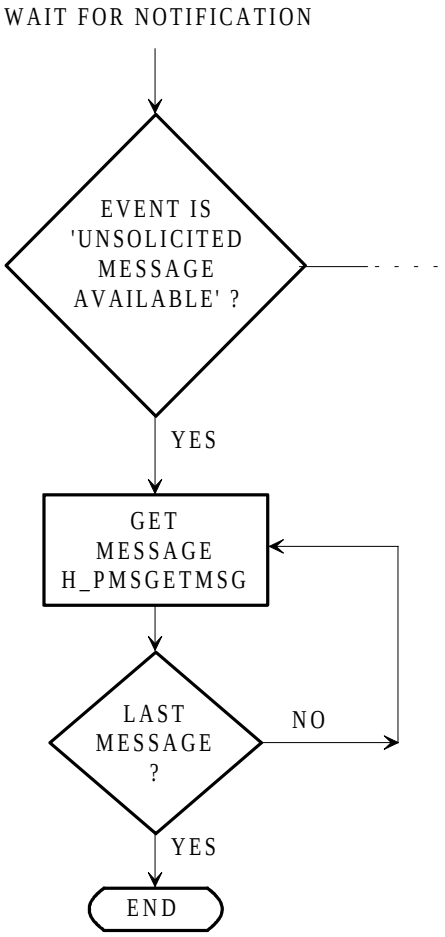

Figure 4-3. Receiving Unsolicited Messages

Each unsolicited message of a certain **class** is stored in a chain of messages related to that class.

The application is notified that an unsolicited message is available each time that a message is placed in an empty chain. This happens for the first message after the creation of a filter set, or for the first message after the application has emptied a chain by receiving all the messages of that class.

When the application is notified that an unsolicited message is waiting, the state of the chain takes the value 'to be delivered' (see the following diagram).

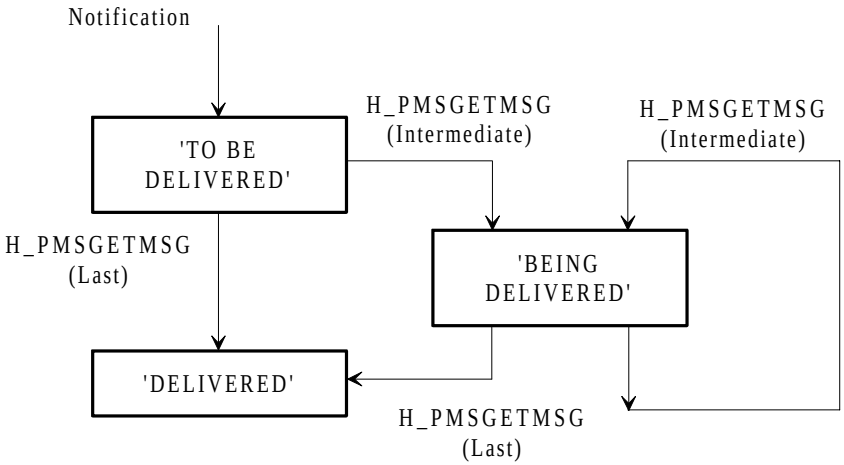

Figure 4-4. State of a Chain of Unsolicited Messages

It can then use the H_PMSGETMSG primitive or function to receive the unsolicited message. The state of the chain becomes 'being delivered', and the message is deleted from the queue file.

While the chain is in this state, the application can use the H_PMSGETMSGFR GPL primitive or C function to get the name of the first unsolicited message, in order to receive the chain again from the beginning.

The next call returns the next message of the same class. When the chain is empty, the state becomes 'delivered'. The next time an unsolicited message of this class is issued, the application will receive a notification.

The application can test for the last two states with the STATE parameter of the H_PMSGETMSG primitive or function.

4.2 How to Declare the DOF 7-PO Objects

The explanations given in the rest of this section are meant as an introduction to the various aspects of coding an application. The examples given generally use only the mandatory parameters of each primitive. For a complete reference to the DOF 7-PO GPL primitive or C functions, and more detailed notes on how to use them, please refer to the sections "The GPL Primitives", and "The C Functions".

The examples in this section are based on extracts of program code. These extracts are taken from the first example application, and are written in GPL. The full listing is provided in Appendix B. Appendix B also contains a listing of the same program written in C.

You must include in your application a declaration of each command, response, and unsolicited message that the application can interpret. (If a response or an unsolicited message can be received, but is not interpreted, that is, you do not intend to use or process the variable fields it contains, you do not need to declare it.)

In GPL, DOF 7-PO uses a single primitive, H_DCPMSXR, to declare both the descriptor and the record for all three types of DOF 7-PO object. The following example shows first the declaration of the descriptor of a command and a response, and then the declaration of the record for the same command and response.

Note that, with the OMH format manager, an unsolicited message is declared in the same way as a response. They are both taken as records of type 'MSG' from the point of view of the H_DCPMSXR primitive.

In the C language, each exchanged structured record is declared in a header file. See later in this section "Notes for C Programmers" for further explanation.

Coding Example in GPL

```
Declaration of descriptor
-----------------------------------------------------------------

$H_DCPMSKDESC ATTRIB = 'BASED(ADDR(L_DESC))'
              PREFIX = 'DESC_'
              ;

$H_DCPMSXR ATTRIB = 'CONSTANT'
           PREFIX = 'CDJ_'
           RECNAME = 'DJ'
           GEN = DESCRIPTOR
           TYPE = CMD
           ;

$H_DCPMSXR ATTRIB = 'CONSTANT'
           PREFIX = 'CSH14_'
           RECNAME = 'SH14'
           GEN = DESCRIPTOR
           TYPE = MSG
           ;
```

Coding Example in GPL

```
Declaration of record
-----------------------------------------------------------------

$H_DCPMSXR ATTRIB = 'BASED(ADDR(L_COMMAND))'
           PREFIX = 'BDJ_'
           RECNAME = 'DJ'
           GEN = RECORD
           TYPE = CMD
           ;

$H_DCPMSXR ATTRIB = 'BASED(ADDR(L_RECORD))'
           PREFIX = 'BSH14_'
           RECNAME = 'SH14'
           GEN = RECORD
           TYPE = MSG
           ;
```

4.2.1 Comments on the Example Declarations

4.2.1.1 Declaration of the Descriptor

The declaration of the descriptor by the application is necessary for all commands, responses, and unsolicited messages to be interpreted by the application.

For the H_DCPMSKDESC primitive, we recommend that you use 'BASED(ADDR(L_DESC))' for the value of the ATTRIB keyword, where L_DESC is a local area large enough to handle the maximum descriptor size. You will then be able to use the names of the fields declared by DCPMSKDESC for every DOF 7-PO object handled by your application (see the description of H_DCPMSKDESC in the next section of the manual).

The maximum descriptor size is 512 bytes. Using this size allows you to reserve enough space to contain the largest descriptor.

For the H_DCPMSXR primitive ATTRIB keyword, we recommend 'CONSTANT' for the descriptor.

RECNAME is the name of the command or message, as it is listed in the *Structured Records* manuals. Messages, that is, both messages issued in response to commands and unsolicited messages, are listed by *key*. Here, the key of the message is SH14.

4.2.1.2 Declaration of the Record

The declaration of the record is necessary for all commands, responses and unsolicited messages to be interpreted by the application, that contain variable fields.

For the value of the H_DCPMSXR primitive ATTRIB keyword, we recommend that you use 'BASED(ADDR(L_COMMAND))' for the record of a command, and 'BASED(ADDR(L_RECORD))' for the record of a response or unsolicited message. L_COMMAND or L_RECORD is a local area large enough to contain the largest command or message record to be handled by the application.

The maximum record size is 2048 bytes. Using this size allows you to reserve enough space to contain the largest record.

4.3 How to Declare the Semaphore Structure

The Programmed Operator application is essentially event-driven. The GCOS 7 semaphore is the mechanism used to signal the arrival of an event.

The semaphore to be used by the application can be declared with the standard GPL primitive H_DCSEM (for the C language equivalent, please refer to Section 6). See also the example program in Appendix B for both GPL and C semaphore declarations.

There are three possible events that can be signalled:

- response available
- unsolicited message available
- DOF 7-PO services (un)available.

The application declares the format of the semaphore message with the H_DCPMSSEMSG GPL primitive or the equivalent C structures. In GPL, the three different events are distinguished by different values of the EVENT parameter, as in the following example.

Full details of this primitive are included in the section "The GPL Primitives".

Coding Example in GPL

```
Declaration of semaphore message
----------------------------------------------------------------


$H_DCPMSSEMMSG PREFIX = 'L_CMD_'
               EVENT = CMD
               ATTRIB = 'BASED(ADDR(L_SEMMSG))'
               ;

$H_DCPMSSEMMSG PREFIX = 'L_MSG_'
               EVENT = MSG
               ATTRIB = 'BASED(ADDR(L_SEMMSG))'
               ;

$H_DCPMSSEMMSG PREFIX = 'L_LNK_'
               EVENT = LNK
               ATTRIB = 'BASED(ADDR(L_SEMMSG))'
               ;
```

4.3.1 Comments on the Example Semaphore Structure Declarations

The EVENT parameter specifies which event the structure is for. The semaphore message contains a **request identifier** (RI), which the application reads to find out which event has been notified.

4.3.1.1 Semaphore Message for Response Available

CMD means "response available". It is associated with the CMDRI request identifier. The message structure is:

```
DCL 1  SEMMSG,
     2 *                CHAR (6),          /* FILLER        */
     2 SEMMSG_CMDID     LOGBIN (16),       /* COMMAND ID.   */
     2 SEMMSG_PMSID     LOGBIN (32),       /* DOF 7-PO IDENT*/
     2 *                BIT (4),           /* FILLER        */
     2 SEMMSG_RI        LOGBIN (12),       /* MAIN RI       */
     2 SEMMSG_SECRI     LOGBIN (16),       /* SECONDARY RI  */
```

4.3.1.2 Semaphore Message for Message Available

MSG means "unsolicited message available". It is associated with the MSGRI request identifier. The message structure is:

```
DCL 1  SEMMSG,
     2 *                CHAR (6),          /* FILLER        */
     2 SEMMSG_FLSID     LOGBIN (16),       /* FILTER SET    */
     2 SEMMSG_PMSID     LOGBIN (32),       /* DOF 7-PO IDENT*/
     2 *                BIT (4),           /* FILLER        */
     2 SEMMSG_RI        LOGBIN (12),       /* MAIN RI       */
     2 SEMMSG_SECRI     LOGBIN (16),       /* SECONDARY RI  */
```

4.3.1.3 Semaphore Message for DOF 7-PO Services (Un)available

LNK means "DOF 7-PO facilities (un)available". It is associated with the LNKRI request identifier. This event gives the result of the request to open a DOF 7-PO connection (see the description of the H_PMSOPEN primitive below). The message structure is:

```
DCL 1  SEMMSG,
     2 *                CHAR (6),          /* FILLER        */
     2 SEMMSG_RC        BIT (16),          /* REASON CODE   */
     2 SEMMSG_PMSID     LOGBIN (32),       /* DOF 7-PO IDENT*/
     2 *                BIT (4),           /* FILLER        */
     2 SEMMSG_RI        LOGBIN (12),       /* MAIN RI       */
     2 SEMMSG_SITE      LOGBIN (16),       /* SITE IDENT.   */
```

4.4 How to Open a DOF 7-PO Connection

DOF 7-PO is a connection-oriented service. A connection is opened as required, and closed when it is no longer needed.

The application uses the GPL primitive or C function, H_PMSOPEN to identify itself and initiate a connection with DOF 7-PO (also called a DOF 7-PO session). The H_PMSCLOSE GPL primitive or C function is used to close the connection.

By default, DOF 7-PO initiates the connection for the user submitting the application. Alternatively it can be initiated for another user by specifying the user name.

The full description of this primitive, and the explanation of all parameters, is included in the section "The GPL Primitives". The example below shows a simple way to initiate a connection, using the two mandatory parameters, and two optional parameters.

Coding Example in GPL

```
Opening a DOF 7-PO connection
-------------------------------------------------------------------


$H_PMSOPEN SEM = ADDR(SEM1)
           USERNAME = 'OPERATOR'
           PASSWORD = 'OP'
           PMSID = L_PMSID
           ;
```

4.4.1 Comments on the Example of Opening a Connection

In this extract from the example program in Appendix B, SEM is the pointer to the semaphore descriptor obtained with the H_DCSEM primitive. This parameter is mandatory; DOF 7-PO must know where to put event notifications for the application.

The connection is to be opened for another user, indicated by USERNAME and PASSWORD (optional).

PMSID is the only output parameter. It is mandatory. It receives the identifier given by DOF 7-PO to the connection. PMSID is then used as an input parameter for all other DOF 7-PO primitives relating to the same connection.

After issuing the H_PMSOPEN primitive, the application waits on the semaphore for the LNK request identifier. When it receives a positive notification that the connection has been made, it can begin to send commands. (Note, however, that the MSG request identifier may arrive before LNK, if the connection is remote, and there are messages waiting.)

Closing a Connection

H_PMSCLOSE has only one mandatory parameter, which is PMSID. See any of the example programs for procedures that use this primitive.

4.5 How to Send a Command

The application uses the H_PMSSENDCMD GPL primitive or C function to submit a command for processing.

The full description of this primitive, and the explanation of all parameters, is included in the section "The GPL Primitives". This extract from the example program shows how to send a command (in this case, CO - Cancel Output) using the mandatory parameters.

Coding Example in GPL

```
Sending a command
-----------------------------------------------------------------

$H_PMSSENDCMD PMSID = L_PMSID
              CMDID = L_CMDID
              DESC = CCO_PMOS_CO_HD
              DESCLN = 'MEASURE(CCO_PMOS_CO_HD)'
              REC = BCO_PMOS_CO_REC
              RECLN = 'MEASURE(BCO_PMOS_CO_REC)'
              ;
```

4.5.1 Comments on the Example of Sending a Command

PMSID is now used as an input parameter to identify the DOF 7-PO connection. It is the identifier assigned by DOF 7-PO to the connection as a result of the execution of the H_PMSOPEN primitive. It is a mandatory parameter.

CMDID is the identifier of the command given by the application. It serves to relate responses to the command. It is mandatory.

The input structure DESC refers to the descriptor of the command, as it was defined with the H_DCPMSXR primitive with the CONSTANT storage attribute (see above "How to declare the DOF 7-PO Objects"). DESCLN is the size of the structure.

REC and RECLN are used for commands which contain variable records. They refer to a local area L_COMMAND, which has been defined to contain the largest command used by the application (see also above "How to declare the DOF 7-PO Objects"). If the record for the command has been declared by the H_DCPMSXR primitive as a structure based by the address of this area, the application will be able to handle all the fields of the record structure.

When you validate a command record, remember that you need give a value only to the significant fields. First reset the presence mask for each region of the record. Then set each byte of the presence mask for a significant field to "01"X (or 0x01 in C language). Then give a value to each significant field.

To validate boolean fields, use the characters "0" for FALSE and "1" for TRUE.

4.6 How to Receive a Response

To receive a response, the application uses the H_PMSGETRP GPL primitive or C function.

The full description of this primitive, and the explanation of all parameters, is included in the section "The GPL Primitives". This extract from the example program shows the simplest way to receive a response. All the parameters shown in this example are mandatory.

Coding Example in GPL

```
Receiving a response
------------------------------------------------------------------

$H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
            CMDID = L_CMD_SEMMSG_CMDID
            MAXDESCLN = 'MEASURE(L_DESC)'
            DESC = L_DESC
            DESCLN = L_DESCLN
            MAXRECLN = 'MEASURE(L_RECORD)'
            REC = L_RECORD
            RECLN = L_RECLN
            ;
```

4.6.1 Comments on the Example of Receiving a Response

PMSID is the identifier of the connection through which the command related to this response was submitted. It is obtained from the semaphore message that notifies the application that the response is available. For the expansion of the semaphore message, see "Comments on the Example Semaphore Structure Declarations" and the full description of the H_PMSGETRP primitive in "The GPL Primitives". PMSID is an input parameter.

CMDID is the identifier of the command given by the application when the command was submitted. It is obtained from the same semaphore message. It is an input parameter.

The output structure referenced by DESC is a work area corresponding to the largest descriptor that the application can receive. MAXDESCLN is an input parameter specifying the maximum size of this area. If the primitive ends normally, DESCLN (output) contains the length of the current descriptor placed in the work area.

The maximum size of a DOF 7-PO descriptor is 512 bytes.

REC, MAXRECLN, and RECLN have exactly the same function in relation to the record currently in the work area.

The maximum value for a DOF 7-PO record length is 2048 bytes.

Other Parameters

Not shown in this example, but recommended if you intend to edit the responses received, are the optional parameters MAXFMPARLN, FMPARPTR, and FMPARLN. They define a structure built by the format manager to contain editing indicators or status information, and which is used internally by DOF 7-PO. We suggest that you define this structure when receiving responses from remote systems, that is when the format manager is DSAC.

4.7 How to Receive an Unsolicited Message

The application can use the H_PMSGETMSG GPL primitive or C function to receive, one at a time, the unsolicited messages belonging to a class of messages defined by a filter set.

The full description of this primitive, and the explanation of all parameters, is included in the section "The GPL Primitives". This extract from the example program shows a simple way to receive a message. All the parameters shown in this example are mandatory, except for STATE (see comments below).

Coding Example in GPL

```
Receiving an unsolicited message
-----------------------------------------------------------------


$H_PMSGETMSG PMSID = L_MSG_SEMMSG_PMSID
             FLSID = L_MSG_SEMMSG_FLSID
             MSGID = L_MSGID
             MAXDESCLN = 'MEASURE(L_DESC)'
             DESC = L_DESC
             DESCLN = L_DESCLN
             MAXRECLN = 'MEASURE(L_RECORD)'
             REC = L_RECORD
             RECLN = L_RECLN
             STATE = L_STATE
             ;
```

4.7.1 Comments on the Example of Receiving an Unsolicited Message

PMSID is the identifier of the connection related to the recipient of this unsolicited message. It is obtained from the semaphore message that notifies the application that the message is available. For the expansion of the semaphore message, see "Comments on the Example Semaphore Structure Declarations".

FLSID is the identifier of the filter set created by the application, or the identifier of the class of specific unsolicited messages. The FLSID is obtained from the semaphore message.

Filters determine whether **generic messages**, that is, the messages issued by the system to the set of **main operators**, are received by a particular operator. The application can create filters for itself using the H_PMSSENDCMD primitive with the CRFLTST and CRFLT commands. See the *System Operator's Guide* for more information about filters.

STATE is an optional output parameter that allows the application to test whether a set of unsolicited messages belonging to the same class has been completely delivered or not. The possible states are explained in the full description of the H_PMSGETMSG primitive in "The GPL Primitives". See also "The Concept of 'State of a Chain'".

Other Parameters

Not shown in this example, but recommended if you intend to edit the messages received, are the optional parameters MAXFMPARLN, FMPARPTR, and FMPARLN. They define a structure built by the format manager to contain editing indicators or status information, and which is used internally by DOF 7-PO. We suggest that you define this structure when receiving messages from remote systems, that is when the format manager is DSAC.

The other parameters in this example are the same as for the H_PMSGETRP primitive (see above "Comments on the Example of Receiving a Response").

4.8 Notes for C Programmers

In the DOF 7-PO application environment, you must declare all commands that will be sent and all responses or unsolicited messages that will be received and interpreted. If a response or an unsolicited message is received but not interpreted, then the declaration of such a message is not necessary.

You declare a message by using the <msg_KEY.h> header file (where KEY is the key of a particular message).

You declare a command by using the <cmd_COMMAND.h> header file (where COMMAND is the name of any DOF 7-PO command.

The declaration of a command named COMMAND requires the inclusion of the <cmd_COMMAND.h> header file and implies:

- The declaration of the descriptor of the command. Its type is named:

  ```
  struct _COMMAND_DESC
  ```

- Before any use of the descriptor, it must be initialized with the following function:

  ```
  { h_init_COMMAND_DESC(descptr) }
  struct _COMMAND_DESC *descptr;
  ```

- The declaration of the record of the command only if the command has some variable fields. Its type is named:

  ```
  struct _COMMAND_REC
  ```

The declaration of a response or an unsolicited message requires the inclusion of the <msg_KEY.h> header file and implies:

- The declaration of the descriptor of the response or the unsolicited message. Its type is named:

  ```
  struct _KEY_DESC
  ```

- Before any use of the descriptor, it must be initialized with the following function:

  ```
  { h_init_KEY_DESC(descptr) }
  struct _KEY_DESC *descptr;
  ```

- The declaration of the record of the response or of the unsolicited message only if the it has some variable fields. Its type is named:

  ```
  struct _KEY_REC
  ```

We recommend the following procedure:

- Declare a `struct _COMMAND_DESC` typed variable for the descriptor of each command used in the application. Initialize it with the `h_init_COMMAND_DESC` function.
- Declare a local area for the command records. This local area must be large enough to contain the largest command record used in the application.
- Declare a record pointer (`struct _COMMAND_REC *`) for each command in the application. This pointer must contain the address of the local area for the command records. Thus, it allows you to reference the command record area with a type dedicated to a particular command.
- Declare a `struct _KEY_DESC` typed variable for the descriptor of each response or unsolicited message which is to be interpreted by the application. Initialize it with the `h_init_KEY_DESC` function.
- Declare a local area for the message records. This local area must be large enough to contain the largest message record used in the application. In this case, the term "message" stands for response or unsolicited message.
- Declare a record pointer (`struct _KEY_REC *`) for each message in the application. This pointer must contain the address of the message area. Thus, it allows you to reference the record area with a type dedicated to each interpreted message.

The maximum size of a descriptor is 512 bytes, and the maximum size of a record is 2048 bytes. These sizes allow you to reserve sufficient areas to contain the largest descriptor and the largest record.

It is also useful for a Programmed Operator application to have a structure for interpreting certain fields of the standard part of a descriptor (which is mapped by the `struct _pms_kdesc` type). For example, the recname field which gives the name of the current record. This structure is a local area large enough to contain the largest descriptor used in the program.

5. The GPL Primitives

This section provides a reference to the GPL primitives that you can use in a Programmed Operator application.

The primitives described are:

H_DCPMSKDESC	Declare DOF 7-PO kernel descriptor
H_DCPMSSEMMSG	Declare DOF 7-PO semaphore message
H_DCPMSXR	Declare DOF 7-PO exchanged record
H_PMSCLOSE	Close a DOF 7-PO connection
H_PMSCVSITE	Convert remote system name
H_PMSEDTMSG	Edit message
H_PMSGETMSG	Get message
H_PMSGETMSGFR	Get message first record name
H_PMSGETRP	Get response
H_PMSGETRPFR	Get response first record name
H_PMSOPEN	Open a DOF 7-PO connection
H_PMSSENDCMD	Send command.

For each primitive you can find information on:

Function
Format
Parameters
Return codes
Comments on usage.

5.1 H_DCPMSKDESC (Declare Kernel Descriptor)

Function

Defines a structure used to map the part of a descriptor that is common to all format managers.

Format

```
$H_DCPMSKDESC  [PREFIX=i-identifier8]
               [ATTRIB=i-char]
                ;
```

Description of parameters

PREFIX	Character string common to all names of the declarative.
ATTRIB	Defines the attributes of the structure (data type, scope, storage, class, etc.).

Comments

This primitive allows an application to access the standard part of a descriptor, which is common to all format managers.

Expansion

```
/* ------  $H_DCPMSKDESC PREFIX=L_;    */

DCL      1  L_HEADER,
 2  L_FMNAME     CHAR(4),
 2  L_DESC_LGTH  FIXED BIN(15),
 2  L_REC_LGTH   FIXED BIN(15),
 2  L_RECNAME    CHAR(8),
 2  L_NATURE,
  3  L_RECTYPE   LOGBIN(2),
  3  L_UMTYPE    LOGBIN(2),
  3  L_MAIN      LOGBIN(1),
  3  L_CKFM      LOGBIN(1),
  3  L_BRIDGE    LOGBIN(1),

  3  L_MBZ       LOGBIN(8),
 2  L_MBZ        CHAR(6),
 2  L_FORMAT     CHAR(4),
 2  L_VERSION    LOGBIN(8),
 2  L_SELECTION,
  3  L_FIELD_NB  LOGBIN(8),
  3  L_MASK_LGTH LOGBIN(8),
  3  L_ZONE_LGTH FIXED BIN(15),
  3  L_TABL_LGTH FIXED BIN(15),
 2  L_MODIFICATION,
  3  L_FIELD_NB  LOGBIN(8),
  3  L_MASK_LGTH LOGBIN(8),
  3  L_ZONE_LGTH FIXED BIN(15),
  3  L_TABL_LGTH FIXED BIN(15),
 2  L_RESPONSE,
  3  L_FIELD_NB  LOGBIN(8),
  3  L_MASK_LGTH LOGBIN(8),
  3  L_ZONE_LGTH FIXED BIN(15),
  3  L_TABL_LGTH FIXED BIN(15);

/*          END OF H_DCPMSKDESC MACRO EXPANSION      */
```

Description of fields

FMNAME	Name of the format manager (OMH, DSAC, ...).
DESC_LGTH	Whole length of the descriptor.
REC_LGTH	Whole length of the related record.
RECNAME	Name of the related record.
NATURE	Nature of the related record. RECTYPE "01"B: unsolicited message. "10"B: command. "11"B: response. UMTYPE significant only for an unsolicited message. "00"B: information message. "01"B: action message to be cancelled by the issuer. "10"B: action message to be acknowledged by the recipient. "11"B: question. MAIN significant only for an unsolicited message. "0"B: specific message. "1"B: generic message. CKFM significant only for the OMH format manager. "0"B: record syntax not checked. "1"B: record syntax checked and correct. BRIDGE significant only for the OMH format manager. "0"B: record sent by means of the DOF 7-PO interface. "1"B: record sent by the OMH bridge.
FORMAT	Name of the parameter format depending on a given format manager (GCOS, AEP, EDIT, ...).
VERSION	Version number of the descriptor.

SELECTION	Description of the parameters of the selection region. FIELD_NB Number of elementary fields contained in the selection region. MASK_LGTH Length of the selection presence mask. ZONE_LGTH Length of the area containing the elementary fields of the selection region. The absent fields are not significant. TABL_LGTH Effective length of the tables describing the elementary fields of the selection region.
MODIFICATION	Description of the parameters of the modification region. FIELD_NB Number of elementary fields contained in the modification region. MASK_LGTH Length of the modification presence mask. ZONE_LGTH Length of the area containing the elementary fields of the modification region. The absent fields are not significant. TABL_LGTH Effective length of the tables describing the elementary fields of the modification region.
RESPONSE	Description of the parameters of the response region. FIELD_NB Number of elementary fields contained in the response region. MASK_LGTH Length of the response presence mask. ZONE_LGTH Length of the area containing the elementary fields of the response region. The absent fields are not significant. TABL_LGTH Length of the tables describing the elementary fields of the response region.

5.2 H_DCPMSSEMMSG (Declare Semaphore-Message)

Function

Defines a structure used to map the semaphore-message for a DOF 7-PO event notified on the DOF 7-PO connection semaphore.

Format

```
$H_DCPMSSEMMSG  [PREFIX=i-identifier8]
                [ATTRIB=i-char]
                 EVENT={CMD|MSG|LNK}
                 ;
```

Description of parameters

PREFIX	Character string common to all names of the declarative.
ATTRIB	Defines the attributes of the structure (data type, scope, storage, class, ...).
EVENT	Defines the DOF 7-PO event notified on the DOF 7-PO connection semaphore.
CMD stands for	"Response available" and is related to the CMDRI request identifier.
MSG stands for	"Unsolicited message available" and is related to the MSGRI request identifier.
LNK stands for	"DOF 7-PO facilities (un)available" and is related to the LNKRI request identifier.

Comments

This primitive allows an application to access the semaphore message for one of the three events notified on the DOF 7-PO connection semaphore. See the next page for the structure of the semaphore messages.

Expansion

```
/* ------  $H_DCPMSSEMMSG PREFIX=L_CMD_ EVENT=CMD;    */

DCL 1  L_CMD_SEMMSG,
     2 *                   CHAR (6),     /* FILLER              */
     2 L_CMD_SEMMSG_CMDID  LOGBIN (16),  /* COMMAND IDENTIFIER  */
     2 L_CMD_SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER */
     2 *                   BIT(4),       /* FILLER              */
     2 L_CMD_SEMMSG_RI     LOGBIN (12),  /* MAIN RI             */
     2 L_CMD_SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI        */

/*          END OF H_DCPMSSEMMSG MACRO EXPANSION      */


/* ------  $H_DCPMSSEMMSG PREFIX=L_MSG_ EVENT=MSG;    */

DCL 1  L_MSG_SEMMSG,
     2 *                   CHAR (6),     /* FILLER              */
     2 L_MSG_SEMMSG_FLSID  LOGBIN (16),  /* FILTER SET IDENT.   */
     2 L_MSG_SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER */
     2 *                   BIT(4),       /* FILLER              */
     2 L_MSG_SEMMSG_RI     LOGBIN (12),  /* MAIN RI             */
     2 L_MSG_SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI        */

/*          END OF H_DCPMSSEMMSG MACRO EXPANSION      */


/* ------  $H_DCPMSSEMMSG PREFIX=L_LNK_ EVENT=LNK;    */

DCL 1  L_LNK_SEMMSG,
     2 *                   CHAR (6),     /* FILLER              */
     2 L_LNK_SEMMSG_RC     BIT (16),     /* REASON CODE         */
     2 L_LNK_SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER */
     2 *                   BIT(4),       /* FILLER              */
     2 L_LNK_SEMMSG_RI     LOGBIN (12),  /* MAIN RI             */
     2 L_LNK_SEMMSG_SITE   LOGBIN (16);  /* SITE IDENTIFIER     */

/*          END   OF H_DCPMSSEMMSG   MACRO   EXPANSION      */
```

Description of fields

SEMMSG_CMDID	Command identifier specified when the command is issued by the H_PMSSENDCMD primitive. Belongs to the semaphore message related to the "Response available" event.

SEMMSG_FLSID	Filter set identifier giving access to a class of messages. For generic messages, this identifier is specified at the creation of a filter set by the CRFLTST command issued by the H_PMSSENDCMD primitive. For specific messages, this identifier has the value "4040"X. Belongs to the semaphore message related to the "Unsolicited message available" event.
SEMMSG_RC	Reason code for the execution of a H_PMSOPEN primitive, returned by the underlying layers (rightmost part of the GCOS 7 return code). Belongs to the semaphore message related to the "DOF 7-PO facilities (un)available" event.
SEMMSG_PMSID	DOF 7-PO connection identifier returned by the H_PMSOPEN primitive when the connection was opened. Belongs to the semaphore message related to all DOF 7-PO events.
SEMMSG_RI	Main request identifier given by the system. It can have two values for DOF 7-PO events: "000"X for "Response available" or "Unsolicited message available" events, "FFF"X for the "DOF 7-PO facilities (un)available" event. Belongs to the semaphore message for all DOF 7-PO events.
SEMMSG_SECRI	Secondary request identifier. It can have the values CMDRI or MSGRI, as specified by the H_PMSOPEN primitive when the connection was opened. Belongs to the semaphore message related to the "Response available" or "Unsolicited message available" events.
SEMMSG_SITE	Internal identifier of the remote system specified at the initiation of the DOF 7-PO connection. A null identifier refers to the local system. Belongs to the semaphore message related to the "DOF 7-PO facilities (un)available" event.

5.3 H_DCPMSXR (Declare Exchanged Record)

Function

Defines a structure (either a descriptor or a record) used to exchange parameters between a service requestor (DOF 7-PO application) and an object manager (DOF 7-PO component).

Format

```
$H_DCPMSXR   RECNAME=i-char8
             GEN={DESCRIPTOR|RECORD}
             OBJ={SYS|NET}
                 ---
             TYPE={CMD|MSG|RSP|USM}
            [PREFIX=i-identifier8]
            [ATTRIB=i-char]
             ;
```

Description of parameters

RECNAME	Name of the structure for which either the descriptor or the record will be generated. This name is the name of a command (for example CJ = Cancel Job) or a key (for example SH14) in the case of a cataloged message (response or unsolicited message). You can look up any RECNAME in the manuals *Structured Records (OMH Format) Part 1 - Commands* and *Structured Records (OMH Format) Part 2 - Messages.*
GEN	Name of the structure for which generation is requested. Possible values are DESCRIPTOR and RECORD. The RECORD structure is declared whenever there are variable fields in the related record.
OBJ	Kind of object for which generation is requested. Possible values are SYS for a system object (which is the default), and NET for a network object.

TYPE	Type of the structure for which generation is requested. For a system object, possible values are CMD for a command, and MSG for a response or an unsolicited message. You can also use RSP for response and USM for an unsolicited message. For a network object, possible values are CMD for a command, RSP or MSG for a response, and USM for an unsolicited message.
PREFIX	Character string common to all names of the declarative.
ATTRIB	Defines the attributes of the structure (data type, scope, storage, class, etc.). It is recommended to use the CONSTANT attribute when the generation of a descriptor is requested, and the BASED attribute when the generation of a record is requested.

Comments

The H_DCPMSXR primitive is used to declare the structures exchanged between the issuer of a piece of information and the recipient(s) of this information.

The exchanged structures may be the following:

- descriptor: structure containing the format manager name, the record name, the parameter format, the version number, and the parameters of each region.
- command: record composed of the presence mask and the parameters of the selection region followed by the presence mask and the parameters of the modification region.
- response: record composed of the presence mask and the parameters of the response region.

The descriptor structure is formed of two parts:

- the first part has a fixed size. It is defined by the expansion of the H_DCPMSKDESC primitive.
- the second part is variable. It consists of three elements that may have a null length. Its whole length is the sum of the lengths of the tables TABL_LGTH describing the selection, modification, and response regions in the structure declared by the H_DCPMSKDESC primitive.

  Each of these elements, if its length is not null, starts with a principal reference table giving the offset of the secondary tables describing the fields of the related region. If there is no secondary table, the related offset is null.

The record structure is formed of one, two, or three regions. Each region is formed of two parts:

- the first part is a presence mask.

  Each field of the selection or modification region corresponds to a byte of the presence mask, set to "00"X if the related field is absent, and set to "01"X if it is significant.

  Each field of the response region corresponds to a bit of the presence mask, set to "0"B if the related field is absent, and set to "1"B if it is significant.

- the second part is formed by the concatenation of all the fields of this region.

5.4 H_PMSCLOSE (Close)

Function

Terminates a DOF 7-PO connection.

Format

```
$H_PMSCLOSE  PMSID=i-lb32
            [SITE=i-char4]
             ;
```

Description of parameters

PMSID	DOF 7-PO connection identifier obtained when the connection is initiated.
SITE	Optional parameter defining the target system if a remote termination is to be performed.

Return codes

Normal

DONE	Termination of connection completed normally.

Abnormal

ARGERR	Error in a parameter.
NOTOPEN	This connection was not initiated.
OPTERR	A remote system was specified without access to the network.
SEQERR	The requestor does not have the right to manage this connection.

Comments

This primitive allows an application to interrupt any activity on the specified connection. The termination of a DOF 7-PO connection may be requested explicitly by the DOF 7-PO requestor, or implicitly by the system at process termination.

The SITE parameter is used to perform a selective termination on the remote connection or connection-relay initiated on the remote target system. Remember that two DOF 7-PO connections are initiated when the H_PMSOPEN is used with the SITE parameter: a local connection on the local system and a remote connection on the target system specified by the SITE parameter. In the same way, a remote connection is implicitly initiated on the remote target system when the H_PMSSENDCMD is used with a SITE parameter different from the target system specified at the initiation of the connection.

A selective termination has no effect on the local connection. However if the system is the system specified at the initiation of the DOF 7-PO connection, the parameter is ignored. In this case, the termination takes effect, on the local connection, as well as on all the remote connection relays.

The following diagram summarizes these points.

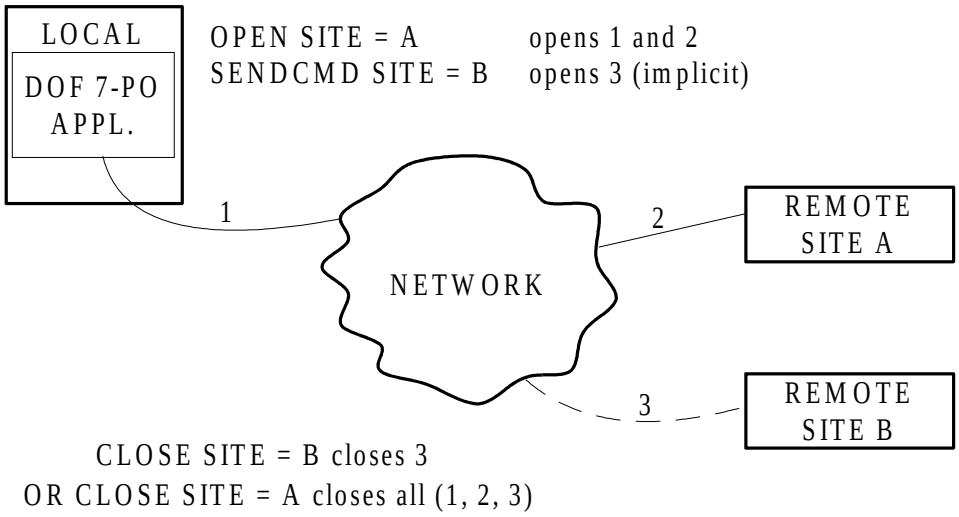

Figure 5-1. Termination with the Site Parameter

The commands and related responses, which the DOF 7-PO requestor has not yet received, are deleted from the DOF 7-PO queue file at the termination of the connection. Repetition of commands submitted with the EVERY parameter is stopped.

Specific information messages are kept in the DOF 7-PO queue file. New specific information messages, sent during the termination of the DOF 7-PO connection of their recipient, are stored and will be delivered the next time the connection concerned is initiated.

Generic information messages, which the recipient has not yet received, are deleted from the queue file. The filter sets are deleted, so no generic messages can be stored during the termination of a DOF 7-PO connection.

As for the **repeatable messages** received by the requestor, they are deleted from the queue file. A DOF 7-PO requestor, which belongs to the generic class of main operators, is removed from the list of recipients in the Repeatable Messages Table. This list will be updated at the next repetition and the unsolicited message possibly sent to other recipients.

The system component issuing a specific question is either aborted or notified with a null reply, depending on the option it uses when the question was issued.

Table 5-1. Summary of the Results of a Termination

nature of the data	treatment of the data
commands/responses	Deleted
specific information messages	Kept
generic information messages	Deleted
repeatable messages	Deleted

5.5 H_PMSCVSITE (Convert Site)

Function

To convert an external system name into the corresponding internal identifier and vice versa.

Format

```
$H_PMSCVSITE    INFORM={INTERNAL|EXTERNAL}
                SITE_IDENT=b_lb16
                SITE_NAME=b_char4
                ;
```

Description of parameters

INFORM	Gives the input format of the conversion.
INTERNAL	Input identification is given by SITE_IDENT.
EXTERNAL	Input identification is given by SITE_NAME.
SITE_IDENT	Area where the internal identifier is given as input or requested as output depending on the INFORM parameter.
SITE_NAME	Area where the external system name is given as input or requested as output depending on the INFORM parameter.

Return codes

Normal

DONE	Normal completion of the conversion.

Abnormal

ARGERR	The identifier given in SITE_IDENT is not associated to an external system name.
NOINIT	The internal tables of the Remote Administrative Exchange Handler (RAEH) have not been created (RAEH server not started).
RSUNKN	The system given in input is unknown in the RAEH tables.

Comments

This primitive is used to convert a DOF 7-PO internal system identifier into the corresponding system name and vice versa. In particular, the internal system identifier, obtained with a "DOF 7-PO facilities (un)available" event, can be converted into the corresponding external system name. In this case, the internal system identifier is contained in the SEMMSG_SITE field of the semaphore message related to the "DOF 7-PO facilities (un)available" event. (See H_DCPMSSEMMSG for a complete description of the semaphore message).

The correspondence between an internal identifier and an external system name is only established for the lifetime of the DOF 7-PO connection on the remote system. That is to say, between the opening of the DOF 7-PO connection on the remote system and the closing of the remote DOF 7-PO connection (with H_PMSCLOSE) or the receipt of an abnormal termination event ("DOF 7-PO facilities unavailable" message).

Therefore, the successful conversion between an internal identifier and an external system name is only guaranteed for this period of time. After the connection has ended, a call to H_PMSCVSITE may be tried but with no guarantee of success.

5.6 H_PMSEDTMSG (Edit Message)

Function

To **edit** a command, a response or an unsolicited message contained in a structured record. In DOF 7-PO, this means to translate the contents of the record into a character string, replacing the variables with values.

Format

```
$H_PMSEDTMSG   DESC=i-structure, DESCLN=i-fb15
              [REC=i-structure, RECLN=i-fb15]
               MESSAGE=o-structure
               | MESSAGE=b-structure
               MAXLN=i-fb15
              [{FMPARPTR=i-ptr, FMPARLN=i-fb15}]
               ;
```

Description of parameters

DESC	Descriptor of the record to be edited, usually declared by the H_DCPMSXR primitive.
DESCLN	Length of the DESC descriptor.
REC	Input record to be edited, usually declared by the H_DCPMSXR primitive. This keyword is used whenever there are variable fields in the related message.
RECLN	Length of the REC record. Giving a null value to RECLN has the same effect as omitting the REC and RECLN keywords.
MESSAGE	Area that will receive the edited record. The output length is returned at the beginning of the area.

```
DCL 1 MESSAGE,
    2  OUTPUT_LENGTH  FIXED BIN(15),
    2  OUTPUT_MESSAGE CHAR(MAXLN);
```

For a network object, the OUTPUT_LENGTH field of the MESSAGE structure is an input-output parameter, which can be used to edit a network administration message. It corresponds to the length of a message item, for example, to one line of a table.

MAXLN	Maximum length of the area declared for the edited message. If the output length exceeds MAXLN, truncation occurs and the TRUNC return code is returned.
FMPARPTR	Pointer to a parameter area provided by the related format manager and obtained with the H_PMSGETRP or H_PMSGETMSG primitives. This area contains control information useful for an edition, and/or status indicators for network objects processed by DSAC.
FMPARLN	Length of the parameter area pointed to by FMPARPTR.

Return codes

Normal

DONE	Function completed normally.

Abnormal

ARGERR	Error in a parameter.
CATERR	The object pointed to by the catalog pointer does not have the characteristics of a catalog.
DAMAGED	Inconsistence in the binary description of a message.
DESCERR	Inconsistency related to the descriptor of the message.
FLNAV	The specified catalog is not available.
LNERR	The length of a message item to edit (OUTPUT_LENGTH of MESSAGE parameter) is greater than the maximum length of the area (MAXLN).
NMTCHERR	The presence mask or a field of the record has a value that is not consistent with the descriptor.
TPUNKN	The message was not found in the catalog.
TRUNC	The length of the edited message exceeds MAXLN. The message is truncated to the value specified by MAXLN.

Comments

The primitive is used to edit a command, response, or unsolicited message according to specified variable parameters given in the REC record. The command, response, or unsolicited message is then stored in the output area.

For system objects, the editing of a cataloged command, response, or unsolicited message is driven by the content of the catalog related to that structure. If the structure contains variable parameters, then the values of these are put in the REC record. If the catalog contains a 'new line' indicator, the two characters carriage return and line feed ("OD25"X) are inserted in the output area to indicate a new line.

For network objects, the OUTPUT_LENGTH field of the MESSAGE parameter defines the length of output lines. The two characters carriage return and line feed ("OD25"X) are inserted in the output area at the end of each line to indicate a new line.

5.7 H_PMSGETMSG (Get Message)

Function

Get an unsolicited message belonging to a class of messages defined by a filter set.

Format

```
$H_PMSGETMSG   PMSID=i-lb32
               FLSID=i-lb16, MSGID=o-lb16
               MAXDESCLN=i-fb15, MAXRECLN=i-fb15
               DESC=o-structure, DESCLN=o-fb15
               REC=o-structure, RECLN=o-fb15
              [CURDESCPTR=i-ptr, NEXTRECNAME=o-char8]
              [MAXFMPARLN=i-fb15
               FMPARPTR=i-ptr, FMPARLN=o-fb15]
              [STATE=o-lb16];
```

Description of parameters

PMSID	Identifier of the DOF 7-PO connection concerned.
FLSID	Identifier of the filter set giving access to an unsolicited message.
MSGID	This parameter receives an unsolicited message identifier when the delivered unsolicited message is a repeatable one. The type of the unsolicited message (action, question, etc.) may be found in the descriptor of the unsolicited message. When the unsolicited message is an information message, this parameter takes the conventional value "FFFF"X.
MAXDESCLN	Maximum size of the area that will receive the descriptor describing the unsolicited message. If the descriptor length exceeds this size, truncation will occur and a WALIM return code will be returned.
MAXRECLN	Maximum size of the area that will receive the unsolicited message(s). If the record length exceeds this size, truncation will occur and a WALIM return code will be returned.
DESC	Descriptor describing the unsolicited message, usually declared by the H_DCPMSXR primitive.

DESCLN	Length of the DESC descriptor.
REC	Record of the unsolicited message, usually declared by the H_DCPMSXR primitive.
RECLN	Length of the REC record.
CURDESCPTR	Pointer to the descriptor of the expected structured record of the current unsolicited message. If this descriptor has a different format from the one sent, a translation can be done by the related format manager.
NEXTRECNAME	Name of the next unsolicited message to be delivered. This parameter allows the application to give, at the next call of the H_PMSGETMSG primitive, a pointer to the descriptor of the expected structured record of this unsolicited message.
MAXFMPARLN	Maximum size of the area that will receive a structure built by the format manager related to the delivered unsolicited message. If the structure length exceeds this size, truncation will occur and a WALIM return code will be returned.
FMPARPTR	Pointer to a parameter area built by the format manager related to the delivered unsolicited message. This area contains control information useful for an edition.
FMPARLN	Current length of the parameter area pointed to by FMPARPTR.

STATE	Parameter giving the state of the delivered piece of information. It specifies also whether any elements of the chain of unsolicited messages have been lost. STATE(1:1) = "1"B: Abnormal termination. STATE(2:4) = Reserved for future use. STATE(6:3) = "001"B: To be delivered. = "010"B: Being delivered. In this case, the return code of the primitive is usually DONE. However DONEIDE or WALIM may override it. = "100"B: Delivered. In this case, the return code of the primitive is usually DATALIM. However DONEIDE or WALIM may override it. STATE(9:8) = "00"X: No element lost. = "01"X: The current element of the chain of unsolicited messages has been delivered but some previous elements have been lost. = "02"X: The current element has been delivered but some following elements have been lost. = "03"X: The current element of the chain of unsolicited messages has been delivered but some previous and some elements following it have been lost.

Return codes

Normal

ALMOST	Function completed normally: other items of the same unsolicited message remain to be delivered. This feature is not used by the current format managers.
DATALIM	Function completed normally: the last unsolicited message has been fully delivered. This class of messages is now empty. In this case, STATE(6:3)="100"B.

DONE	Function completed normally: the current unsolicited message has been fully delivered. Some unsolicited messages belonging to this class remain. In this case, STATE(6:3)="010"B.
DONEIDE	Unsolicited message delivered, but the translation requested by setting the CURDESCPTR parameter cannot be done. The unsolicited message is delivered as it was sent.
WALIM	The unsolicited message has been delivered but truncation has occurred because the output length exceeds MAXDESCLN, MAXRECLN, or MAXFMPARLN.
Abnormal	
ARGERR	Error in a parameter.
DESCERR	Inconsistency related to the descriptor of the unsolicited message.
EXHAUST	The last unsolicited message of this class has already been delivered.
INDUNKN	Incorrect value of the filter set identifier.
ITMNAV	The DOF 7-PO requestor has not been notified of the availability of an unsolicited message related to this class.
NOTOPEN	This connection was not initiated.
SEQERR	The requestor does not have the right to manage this connection.
TOOLATE	The previously available unsolicited message is no longer significant because it is a repeatable message already taken into account by another recipient. No other message of this class is available.

Comments

This primitive is used to obtain, one by one, the unsolicited messages sent to a DOF 7-PO requestor.

The identifier of the class of the required unsolicited messages is given in the FLSID parameter. The blank identifier ("4040"X) is the conventional identifier for specific messages. On the other hand, the name of the filter set giving access to the message is the identifier for generic messages.

The filters are created by the CRFLTST and CRFLT commands, which can be submitted from an application by means of the H_PMSSENDCMD primitive. The name of a filter set defines the name of a class of messages. A specific message is therefore always sent to its recipients, whether the connection is initiated or not, while a generic message is sent to the requestor(s) with filters matching the unsolicited message.

The semaphore message related to the "Unsolicited message available" event is declared by the H_DCPMSSEMMSG primitive and is defined as follows:

```
DCL 1  SEMMSG,
     2               CHAR (6),     /* FILLER                */
     2 SEMMSG_FLSID  LOGBIN (16),  /* FILTER SET IDENT.     */
     2 SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER   */
     2 *             BIT(4),       /* FILLER                */
     2 SEMMSG_RI     LOGBIN (12),  /* MAIN RI               */
     2 SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI          */
```

SEMMSG_FLSID is the filter set identifier giving access to the class of messages and SEMMSG_PMSID is the DOF 7-PO connection identifier. SEMMSG_RI has the value "000"X. SEMMSG_SECRI is the request identifier specified at the initiation of the connection and related to the "Unsolicited message available" event.

The notification on the semaphore given at the initiation of the DOF 7-PO connection is performed only when an unsolicited message is stored in an empty class of messages, (i.e., the first time after the creation of a filter set, or each time after all unsolicited messages have been received by an application). The state of the class of messages then takes the value "001"B; i.e., to be delivered.

When the H_PMSGETMSG primitive is called, the system returns an unsolicited message corresponding to the specified identifier (the field STATE(6:3) takes the value "010"B; i.e., being delivered, and deletes the unsolicited message from the queue file, so that the next call returns the next unsolicited message of the same class. When the class becomes empty, a special return code is given back with the last unsolicited message and the STATE(6:3) field takes the value "100"B; i.e., delivered. The next issuing of an unsolicited message belonging to this class will give rise to a notification on the connection semaphore.

The unsolicited message (i.e., the descriptor and the related record) is returned by default to the requestor as it was sent by the issuer. However the requestor has the possibility of knowing, before getting the unsolicited message, the name of the record that will be obtained on the next call. The name of the first record is given by the H_PMSGETMSGFR primitive. Then each H_PMSGETMSG primitive gives the name of the next record. This name is meaningless for the last unsolicited message of the class, it is set to "space". The DOF 7-PO requestor can thus specify, by the CURDESCPTR parameter, the descriptor describing the structured record of the unsolicited message that it wants to receive.

The object manager and the requestor may have different version numbers. The access method ensures compatibility in the exchanged structures so that the two DOF 7-PO service users understand each other.

The applications may receive, optionally, a complementary structure built by the related format manager. This structure contains the filtering criteria in the case of the OMH format manager, or editing indicators or status information for the DSAC format manager. We recommend that you specify this structure when using DSAC.

The output areas are not modified if the function is not successfully completed.

Note on the STATE parameter. When the primitive delivers a response, and the return code is normal, it is still often necessary to use the STATE parameter. You may want to know:

1. If the response is truncated or not (WALIM)
2. If the translation requested was done or not (DONEIDE)
3. If the response is the last one (DATALIM)
4. If any responses were lost during delivery.

Since the return code cannot contain all this information at the same time, the priority for reporting the information is the order of the list above. In order to process items 3 and 4, it is therefore necessary to look at the STATE field, where this information is always available.

5.8 H_PMSGETMSGFR (Get Message First Record Name)

Function

Get the record name of the first unsolicited message of the specified class.

Format

```
$H_PMSGETMSGFR   PMSID=i-lb32
                 FLSID=i-lb16
                [FORMAT={"GCOS"|"AEP"}]
                          ---
                 RECNAME=o-char8
                 ;
```

Description of parameters

PMSID	Identifier of the DOF 7-PO connection concerned.
FLSID	Identifier of the filter set giving access to an unsolicited message.
FORMAT	Expected format of the record related to the unsolicited message.
RECNAME	Name of the record related to the next unsolicited message to be delivered corresponding to the named filter set.

Return codes

Normal

DONE	Function completed normally.

Abnormal

ARGERR	Error in a parameter.
EXHAUST	The last unsolicited message of this class has already been delivered.
INDUNKN	Incorrect value of the filter set identifier.
ITMNAV	The DOF 7-PO requestor has not been notified of the availability of an unsolicited message related to this class.

NOTOPEN	This DOF 7-PO connection was not initiated.
SEQERR	The requestor does not have the right to manage this connection.
TOOLATE	The previously available unsolicited message is no longer significant because it is a repeatable message already taken into account by another recipient. No other unsolicited message of this class is available.

Comments

This primitive allows the requestor to get the record name of the first unsolicited message related to a class of messages specified by the PMSID, FLSID parameters. These parameters were obtained from the semaphore message at the time of the notification of the "Unsolicited message available" event.

The primitive provides a simple way of knowing in advance the identifier of the first available unsolicited message.

It can also be used to go back to the first unsolicited message of the class in order to read the chain of messages again from the beginning.

The semaphore message related to the "Unsolicited message available" event is declared by the H_DCPMSSEMMSG primitive and is defined as follows:

```
DCL 1  SEMMSG,
     2 *             CHAR (6),     /* FILLER               */
     2 SEMMSG_FLSID  LOGBIN (16),  /* FILTER SET IDENT.    */
     2 SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER  */
     2 *             BIT(4),       /* FILLER               */
     2 SEMMSG_RI     LOGBIN (12),  /* MAIN RI              */
     2 SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI         */
```

SEMMSG_FLSID is the filter set identifier giving access to the class of messages and SEMMSG_PMSID is the DOF 7-PO connection identifier. SEMMSG_RI has the value "000"X. SEMMSG_SECRI is the request identifier specified at the initiation of the DOF 7-PO connection and related to the "Unsolicited message available" event.

In the case of an unsolicited message related to the DSAC format manager, the expected format of the records may be GCOS or AEP. If the expected format is the GCOS format, then the RECNAME field is filled with the CODE abbreviation and the CLASS abbreviation; the rest of the field is padded with spaces. If the expected format is the AEP format, then the RECNAME field is filled with the CLASS number and the CODE number (both 8-bit numbers); the rest of the field is padded with binary zeroes.

5.9 H_PMSGETRP (Get Response)

Function

Get a response, or an element of a response, to a command.

Format

```
$H_PMSGETRP    PMSID=i-lb32
               CMDID=i-lb16
               MAXDESCLN=i-fb15, MAXRECLN=i-fb15
               DESC=o-structure, DESCLN=o-fb15
               REC=o-structure, RECLN=o-fb15
              [CURDESCPTR=i-ptr, NEXTRECNAME=o-char8]
              [MAXFMPARLN=i-fb15
               FMPARPTR=i-ptr, FMPARLN=o-fb15]
              [STATE=o-lb16]
               ;
```

Description of parameters

PMSID	Identifier of the DOF 7-PO connection concerned.
CMDID	Identifier of the command.
MAXDESCLN	Maximum size of the area that will receive the descriptor describing the response. If the descriptor length exceeds this size, truncation will occur and a WALIM return code will be returned.
MAXRECLN	Maximum size of the area that will receive the response(s). If the record length exceeds this size, truncation will occur and a WALIM return code will be returned.
DESC	Descriptor describing the response, usually declared by the H_DCPMSXR primitive.
DESCLN	Length of the DESC descriptor.
REC	Record of the response, usually declared by the H_DCPMSXR primitive.
RECLN	Length of the REC response.

CURDESCPTR	Pointer to the descriptor of the expected structured record of the current response. If this descriptor has a different format from the one sent, a translation can be done by the related format manager.
NEXTRECNAME	Name of the next response delivered. This parameter allows the application to give on the next call of the H_PMSGETRP primitive a pointer to the descriptor of the expected structured record of this response.
MAXFMPARLN	Maximum size of the area that will receive a structure built by the format manager concerned and related to the delivered response. If the structure length exceeds this size, truncation will occur and a WALIM return code will be returned.
FMPARPTR	Pointer to a parameter area built by the format manager concerned and related to the delivered response. This area contains control information useful for editing the response.
FMPARLN	Current length of the parameter area pointed to by FMPARPTR.
STATE	Parameter giving the state of the delivered piece of information. It specifies also whether any elements of the chain of responses have been lost. STATE(1:1) = "1"B: Abnormal termination. The limit for the chain command/response has been reached. However, it is still possible to read all the elements of the chain. STATE(2:4) = Reserved for future use. STATE(6:3) = "001"B: To be delivered. = "010"B: Being delivered. In this case, the return code of the primitive is usually DONE. However DONEIDE or WALIM may override it. = "100"B: Delivered. In this case, the return code of the primitive is usually DATALIM. However DONEIDE or WALIM may override it.

STATE(9:8)
= "00"X: No element lost.
= "01"X: The current element of the chain of responses has been delivered but some previous elements have been lost.
= "02"X: The current element has been delivered but some of the elements following it have been lost.
= "03"X: The current element of the chain of responses has been delivered but some previous and some following elements have been lost.

Return codes

Normal

ALMOST	Function completed normally: other items of the same response remain to be delivered. This feature is not used by the current format managers.
DATALIM	Function completed normally: the last response has been fully delivered. No responses remain related to this command. In this case, STATE(6:3)="100"B.
DONE	Function completed normally: the current response has been fully delivered. There remain some responses related to this command. In this case, STATE(6:3)="010"B.
DONEIDE	Response delivered, but the translation requested by setting the CURDESCPTR parameter cannot be done. The response is delivered as it was sent.
WALIM	The response has been delivered but truncation has occurred because the output length exceeds MAXDESCLN, MAXRECLN, or MAXFMPARLN.

Abnormal

ARGERR	Error in a parameter.
DESCERR	Inconsistency related to the descriptor of the response.
EXHAUST	The last response related to this command has already been delivered.
INDUNKN	Incorrect value of the command identifier.

ITMNAV	The response is not available. The semaphore of the DOF 7-PO connection has not been notified.
NOTOPEN	This DOF 7-PO connection was not initiated.
SEQERR	The requestor does not have the right to manage this connection.

Comments

This primitive is used to obtain, one by one, the elementary responses related to a given command specified by the command identifier.

Every time an object manager sends a response to a given command, this response is chained into the DOF 7-PO queue file and linked to the command. When the last response is sent, the submitter of the command is notified on the semaphore given at the initiation of the connection.

The semaphore message related to the "Response available" event is declared by the H_DCPMSSEMMSG and is defined as follows:

```
DCL 1  L_CMD_SEMMSG,
     2 *                  CHAR (6),     /* FILLER              */
     2 L_CMD_SEMMSG_CMDID  LOGBIN (16),  /* COMMAND IDENTIFIER  */
     2 L_CMD_SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER */
     2 *                  BIT(4),       /* FILLER              */
     2 L_CMD_SEMMSG_RI     LOGBIN (12),  /* MAIN RI             */
     2 L_CMD_SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI        */
```

SEMMSG_CMDID is the command identifier and SEMMSG_PMSID is the DOF 7-PO connection identifier. SEMMSG_RI has the value "000"X. SEMMSG_SECRI is the request identifier specified at the initiation of the connection and related to the "Response available" event.

The notification on the semaphore given at the initiation of the connection is done only when the last response has been stored in the DOF 7-PO queue file. When the H_PMSGETRP primitive is called, the system returns the current elementary response corresponding to the specified identifier, and the queue pointer to the response to be delivered is updated so that the next call returns the next response of the same chain. When the last response is obtained, a special return code is passed with it, and the whole chain (i.e., the command and the related responses) is deleted from the queue file.

The response (i.e., the descriptor and the related record) is returned by default to the DOF 7-PO requestor as it was sent by the issuer. However the DOF 7-PO requestor has the possibility of knowing, before getting the response, the name of the record that will be obtained on the next call. The name of the first record is given by the H_PMSGETRPFR primitive. Then each H_PMSGETRP primitive gives the name of the next record. This name is meaningless for the last response of the chain, it is set to "space". The requestor can thus specify, by the CURDESCPTR parameter, the descriptor describing the structured record of the response it wants to receive.

The object manager and the requestor may have different version numbers. This access method ensures compatibility in the exchanged structures so that the two DOF 7-PO service users understand each other.

As long as a response is not fully delivered, the requestor has the possibility of obtaining the first elementary response again, by using the H_PMSGETRPFR primitive to move the current pointer to the first response of the chain, and then by using the H_PMSGETRP primitive to receive the response.

The applications can optionally receive a complementary structure built by the related format manager. This complementary structure can be used when the response is edited, for example. We recommend that you specify this output structure when you use DSAC. It contains information about the normal or abnormal execution of the commands as well as indicators needed when using the H_PMSEDTMSG primitive.

The output areas are not modified if the function is not completed successfully.

Note on the STATE parameter. When this primitive returns a response and the return code is normal, it is still often necessary to use the STATE parameter. You may want to know the following information:

1. If the response has been truncated or not (WALIM)
2. If the translation requested was done or not (DONEIDE)
3. If the response is the last (DATALIM)
4. If any responses have been lost during delivery.

Since the return code cannot contain all these items, DOF 7-PO uses the priority shown in this list. To process items 3 and 4 above, the program should examine the STATE parameter, where this information is always available.

5.10 H_PMSGETRPFR (Get Response First Record Name)

Function

Get the record name of the first response of the specified command.

Format

```
$H_PMSGETRPFR   PMSID=i-lb32
                CMDID=i-lb16
                [FORMAT={"GCOS"|"AEP"}]
                        ---
                RECNAME=o-char8
                ;
```

Description of parameters

PMSID	Identifier of the DOF 7-PO connection concerned.
CMDID	Identifier of the command.
FORMAT	Expected format of the record related to the response.
RECNAME	Name of the record related to the first delivered response corresponding to the named command/responses chain.

Return codes

Normal	
DONE	Function completed normally.
Abnormal	
ARGERR	Error in a parameter.
INDUNKN	Incorrect value of the command identifier.
ITMNAV	The response is not available. The connection semaphore has not been notified.
NOTOPEN	This connection was not initiated.
SEQERR	The requestor does not have the right to manage this connection.

Comments

This primitive allows the requestor to get the record name of the first response associated to a command specified by the PMSID, CMDID parameters. These parameters were obtained from the semaphore message at the time of notification of the "Response available" event.

The primitive provides a simple way of knowing in advance the identifier of the first available response.

The semaphore message related to the "Response available" event is declared by the H_DCPMSSEMMSG and is defined as follows:

```
DCL 1  SEMMSG,
     2 *             CHAR (6),     /* FILLER               */
     2 SEMMSG_CMDID  LOGBIN (16),  /* COMMAND IDENTIFIER   */
     2 SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER  */
     2 *             BIT(4),       /* FILLER               */
     2 SEMMSG_RI     LOGBIN (12),  /* MAIN RI              */
     2 SEMMSG_SECRI  LOGBIN (16);  /* SECONDARY RI         */
```

SEMMSG_CMDID is the command identifier and SEMMSG_PMSID is the DOF 7-PO connection identifier. SEMMSG_RI has the value "000"X. SEMMSG_SECRI is the request identifier specified at the initiation of the DOF 7-PO connection and related to the "Response available" event.

This primitive has another effect. While a response is not fully delivered, the requestor has the possibility of obtaining the first elementary response again, by using the H_PMSGETRPFR primitive to move the current pointer to the first response of the chain, and then by using the H_PMSGETRP primitive to receive the response.

In the case of a response related to the DSAC format manager, the expected format of the records may be GCOS or AEP. If the expected format is the GCOS format, then the RECNAME field is filled with the CODE abbreviation and the CLASS abbreviation; the rest of the field is padded with spaces. If the expected format is the AEP format, then the RECNAME field is filled with the CLASS number and the CODE number (both 8-bit numbers); the rest of the field is padded with binary zeroes.

5.11 H_PMSOPEN (Open)

Function

Initiate a DOF 7-PO connection for a DOF 7-PO requestor.

Format

```
$H_PMSOPEN  SEM=i-ptr PMSID=o-lb32
           [PMSNB={0|i-lb16}]
           [MSGRI={1|i-lb16}] [MSGPRTY={15|i-fb15}]
           [CMDRI={2|i-lb16}] [CMDPRTY={15|i-fb15}]
                              [LNKPRTY={15|i-fb15}]
           [WAIT] [CLEAN] [REPEATMSG]
          [[USERNAME=i-char12  PASSWORD=i-char12]
           [PROJECT=i-char12] [BILLING=i-char12]
           [STATION=i-char8]]
          [{SITE=i-char4
          |FORCE}]
           ;
```

Description of parameters

SEM	Pointer to the semaphore descriptor of the semaphore on which the "Response available", "Unsolicited message available" and / or "DOF 7-PO facilities (un)available" events will be notified.
PMSID	Identifier of the DOF 7-PO connection given by the system. An application can initiate several DOF 7-PO connections simultaneously, so the PMSID allows it to specify the connection on which the primitive is to be submitted. If the initiation is not completed, the PMSID parameter is filled with the conventional value "FFFFFFFF"X.
PMSNB	This optional parameter specifies the number of the DOF 7-PO connection for this user. The USERNAME-PMSNB pair must be unique in the system: this pair fully identifies a connection. By default, the DOF 7-PO kernel gives it the number 0. The range of allowed values for this parameter is 0-15.

MSGRI	Request identifier which will be returned in the semaphore message at the time of the notification of the "Unsolicited message available" event on the SEM semaphore.
MSGPRTY	Queueing priority of the "Unsolicited message available" notification on the previous semaphore (0 to 15, the lowest value having the highest priority). It is used to define a hierarchy among various events notified on the same semaphore. If the given priority is outside the two bounds, this priority is forced to 15. The default value is 15.
CMDRI	Request identifier which will be returned in the semaphore message at the time of the notification of the "Response available" event on the SEM semaphore.
CMDPRTY	Queueing priority of the "Response available" notification on the previous semaphore (0 to 15, the lowest value having the highest priority). It is used to define a hierarchy among various events notified on the same semaphore. If the given priority is outside the two bounds, this priority is forced to 15. The default value is 15.
LNKPRTY	Queueing priority of the "DOF 7-PO facilities (un)available" notification on the previous semaphore (0 to 15, the lowest value having the highest priority). It is used to define a hierarchy among various events notified on the same semaphore. If the given priority is outside the two bounds this priority is forced to 15. The default value is 15.
WAIT	This self-identifying value is used to request a wait on a semaphore in the case of unavailability of the DOF 7-PO kernel. The H_PMSOPEN primitive ends when the kernel is available and the connection is initiated. On the one hand, DOF 7-PO, which is implemented as a server, is unavailable before SYSTEM READY. On the other hand, after a system crash RERUN may restart DOF 7-PO applications before the DOF 7-PO server. A H_PMSOPEN primitive can therefore be submitted before DOF 7-PO is available. This keyword allows the user to handle these possibilities.

CLEAN	This self-identifying value is used to purge all unsolicited messages waiting for this DOF 7-PO requestor at the initiation of the connection. This purge applies only to the specific messages since the generic messages have been purged at the previous termination of the connection (if there were any). The default option is to notify the requestor if unsolicited messages are available.
REPEATMSG	This self-identifying value allows repetition of the repeatable messages when they have been received by the requestor but not yet been acknowledged or replied to. If this keyword is not specified at the initiation of the connection, the repeatable messages are sent to this requestor only once during the connection.
USERNAME	This optional parameter is used to give the identity of the DOF 7-PO requestor. The default value is the submitter of the application. This value is checked against the SITE.CATALOG.
PASSWORD	This parameter is mandatory when USERNAME is present. It specifies the password related to this user name and is checked against the SITE.CATALOG. If no password has been defined in the catalog, this field is set to spaces.
PROJECT	This parameter is used to modify the implicit project related to the current user.
BILLING	This parameter is used to modify the implicit billing related to the current project.
STATION	This optional parameter is used to change the implicit station related to the current project. This station is used to route the outputs of the jobs submitted during the connection.
SITE	This optional parameter is used to initiate a DOF 7-PO connection on the specified system. By default, all commands submitted during this connection will be sent to this specified system.
FORCE	This keyword is used to initiate a DOF 7-PO connection on a DOF 7-PO session that has been opened on the local system by a remote requestor.

Return codes

Normal	
DONE	Function completed normally.
Abnormal	
ARGERR	Error in a parameter.
ENTRYOV	No more entries available in the system tables. 16 requestors having the "MAIN" right are already connected.
NOINIT	The DOF 7-PO kernel is unavailable.
PROJUNKN	Project unknown (not found in catalog).
PSWVIOL	Password violation.
RLBPUNKN	Relation Billing/Project unknown (not found in catalog).
RSUNKN	Remote system unknown.
SHUTDOWN	GCOS 7 is in the shutdown phase, so new connections are refused.
STTNUNKN	Station unknown (not found in catalog).
USERUNKN	User unknown (not found in catalog).
WITHSYN	The USERNAME - PMSNB pair has a synonym in the system tables.

Comments

The use of this primitive allows an application to identify itself to the system and to initiate a connection with the DOF 7-PO kernel. By default, the connection is initiated for the submitter of the job, otherwise it is initiated for the user whose name is specified.

Any user known by the system catalog may initiate a DOF 7-PO connection. The default project, billing, or station can be replaced by another, if specified. The system deduces from this identification the access rights to the system commands.

NOTE:

No more than 16 requestors having the "MAIN" right may be connected at the same time.

The same user can simultaneously initiate sixteen DOF 7-PO connections by qualifying them with a DOF 7-PO connection number. The "user name - DOF 7-PO connection number" pair must be unique in the system: it is this pair which completely identifies the requestor. Be careful if you have not explicitly specified a user name at connection time.

The system returns a DOF 7-PO identifier, which must be used later with all primitives, related to this connection.

Up to 5000 DOF 7-PO connections may be initiated on a system at the same time. These include the local and the remote connections for both the interactive and the batch requestors.

The requestor specifies at the initiation of a connection the address of a semaphore on which the "Response available", "Unsolicited message available" or "DOF 7-PO facilities (un)available" events will be notified. The requestor also specifies the Request Identifiers which will be returned in the semaphore message at the time of the notification of the "Response available", "Unsolicited message available" or "DOF 7-PO facilities (un)available" events, and the priorities related to these notifications.

Some checks are performed when a connection is initiated. As these checks can go on a long time for a remote system, initiation is always asynchronous. When the local checks are done, the application receives a return code, which is DONE if the checks are successful, but command submission is not yet allowed. When the connection is fully established, a notification is sent on the semaphore related to the DOF 7-PO connection.

The semaphore message related to the "DOF 7-PO facilities (un)available" event is declared by the H_DCPMSSEMMSG and is defined as follows:

```
DCL 1  SEMMSG,
     2 *             CHAR (6),     /* FILLER                */
     2 SEMMSG_RC     BIT (16),     /* REASON CODE           */
     2 SEMMSG_PMSID  LOGBIN (32),  /* DOF 7-PO IDENTIFIER   */
     2 *             BIT(4),       /* FILLER                */
     2 SEMMSG_RI     LOGBIN (12),  /* MAIN RI               */
     2 SEMMSG_SITE   LOGBIN (16);  /* SITE IDENTIFIER       */
```

SEMMSG_RC is the reason code of the execution and SEMMSG_PMSID is the DOF 7-PO session identifier. SEMMSG_RI is the request identifier specified at the initiation of the DOF 7-PO session and related to the "DOF 7-PO facilities (un)available" event. The only possible value is "FFF"X. SEMMSG_SITE is the internal identifier associated to the system specified at the initiation of the connection. A null identifier corresponds by convention to the local system.

If the DOF 7-PO kernel is not available when the H_PMSOPEN primitive is called, then an abnormal NOINIT return code is returned unless the WAIT keyword has been used with the H_PMSOPEN primitive. In this case, the application is set in a wait state until the kernel is available again.

When the kernel becomes unavailable, outstanding requests are aborted with an abnormal return code and new requests are rejected. The application must in this case ask for another DOF 7-PO connection by means of the H_PMSOPEN primitive, possibly with the WAIT parameter.

Several quotas are allocated to each DOF 7-PO connection in order to avoid an overflow of the queue file through overuse by a single user. These quotas are defined by the maximum number of outstanding objects handled during a DOF 7-PO connection and by the size of each object. The maximum number of outstanding commands is set to 32. The maximum number of outstanding filter sets is set to 16. The whole size of a command chain (i.e., a chain of a command and its related responses) is set to 512K. The whole size of a chain of unsolicited messages, related to a given filter set or a chain of specific messages, is set to 64K.

To give an idea of the capacity of a 64 Kbyte chain, this size can store 128 JB08 unsolicited messages.

When the quota related to the number of commands is reached any further command submission is rejected until the number of outstanding commands goes below the maximum allowed. When the quota related to the size of a handled chain is reached, any issuing of response or unsolicited message is rejected until the chain is cleared by the application.

The optional CLEAN parameter is used to purge the chain of specific messages at the initiation of the connection. If the CLEAN option is not used, and this is the default mode, the application is notified on its connection semaphore if there are any unsolicited messages available. This notification may arrive before the "DOF 7-PO facilities (un)available" notification. In this case, the application can retrieve the messages immediately.

The optional REPEATMSG parameter allows an application to receive the repeatable messages at each repetition. By default the application receives the unsolicited message only once.

The optional SITE parameter is used to define the system on which all commands are submitted by default. If this parameter is not specified, the default system is the local one. If the specified system is a DPS7000 system supporting DOF 7-PO and it is different from the local system, a second DOF 7-PO connection called connection-relay is initiated on the target system through the medium of the Network Exchange Handler. The Remote Administrative Exchange Handler (RAEH server) must be started when initiating a DOF 7-PO connection on a remote system.

Reconfiguration

If a session that has been opened with a remote system fails, or is closed, any specific unsolicited messages not yet delivered are kept. These messages can be recovered by opening another session with H_PMSOPEN, specifying the same PMSNB that was used for the session when it was opened previously.

5.12 H_PMSSENDCMD (Send Command)

Function

Send a command to a system or telecommunication object manager.

Format

```
$H_PMSSENDCMD  PMSID=i-lb32, CMDID=i-lb16
               DESC=i-structure, DESCLN=i-fb15
              [REC=i-structure, RECLN=i-fb15]
              [STATION=i-char8]
              [EVERY=i-fb15]
              [SITE=i-char4]
              [NOLOG={"NO" | "YES" | "NOLOG"}]
               ;
```

Description of parameters

PMSID	Identifier of the DOF 7-PO connection concerned.
CMDID	Identifier of the command. This parameter is used to handle the command and its related responses.
DESC	Descriptor of the command record, usually declared by the H_DCPMSXR primitive.
DESCLN	Length of the command descriptor.
REC	Record containing the parameters of the command, usually declared by the H_DCPMSXR primitive. This keyword is used every time that there are variable fields in the related command.
RECLN	Length of the command record. Giving a null value to RECLN has the same effect as omitting both keywords REC and RECLN.
STATION	Name of the logical station specified in the command. This parameter is ignored if the format of the record is not the GCOS format. This station modifies the semantics of the command.

EVERY	Optional parameter which is used to submit the command at regular time intervals. This interval is given in seconds by the value of the EVERY parameter. The -1 value means by convention that no repetition is required. A response to a repeatable command, which is not being delivered, is refreshed by the following response as soon as it is available. A DOF 7-PO requestor is thus assured of receiving the latest response.
SITE	Optional parameter that is used to submit a command on a target system different from the system specified at connection initiation time.
NOLOG	Optional parameter which can be used to cause the reply to the command to be placed in the SYS.LOGC file. A value of "NO" (the default) results in this happening: a value of "YES" or "NOLOG" means that this is not done. Note that this parameter has no effect unless the DOF 7-PO session is opened for a MAIN operator.

Return codes

Normal	
DONE	Function completed normally.
Abnormal	
ARGERR	Error in a parameter.
ARVIOL	Command not accessible in this environment.
CDERR	The command structure is incompatible with the descriptor that describes it.
CDUNKN	Unknown command.
DESCERR	Inconsistency related to the descriptor of the command.
NMTCHERR	The command record is not validated as needed by the syntax described in the command descriptor.
NOTOPEN	This DOF 7-PO connection was not initiated.
PROJUNKN	Project unknown (not found in catalog).

PSWVIOL	Password violation.
RLBPUNKN	Relation Billing/Project unknown (not found in catalog).
RSUNKN	Remote system unknown.
SEQERR	The requestor does not have the right to manage this connection.
SHUTDOWN	GCOS 7 is in the shutdown phase so the issuing of new commands is denied.
STTNUNKN	Station unknown (not found in catalog).
SYSOV	The command cannot be taken into account because of either: an overflow on the number of outstanding commands (restricted to 32), or a space overflow in the file used by DOF 7-PO to store the chains of commands and their related responses.
USERUNKN	User unknown (not found in catalog).
WAOV	Overflow on the work area used to communicate with the DOF 7-PO server. The descriptor and the record of the command are too large.
WITHSYN	The request identifier; i.e., the PMSID/CMDID pair has a synonym.

Comments

This primitive is used to submit a command to a system belonging to the primary network. This command must belong to the current environment of the DOF 7-PO requestor. The access rights are derived from those of the user defined at the initiation of the connection. This allows the possibility of submitting a command for another user than the owner of the current step.

The submitter must associate an identifier to the command. This command identifier must be different from all identifiers related to the outstanding commands belonging to the same connection. The command identifier allows the submitter to handle the command and its related responses. By default, a command is submitted on the system defined at the initiation of the DOF 7-PO connection. An optional parameter is used to modify this default system. If the target system is different from the local system, a connection is created on the remote system if necessary (i.e. if the remote system is a DPS 7000 system supporting DOF 7-PO and if no command had yet been sent to this remote system by this connection). A DOF 7-PO connection is created on the remote site with the three parameters PMSNB, USERNAME, and PASSWORD. The other parameters PROJECT, BILLING, etc. are found by default in the system catalog of the remote site, for the specified user, unless they were previously supplied via the H_PMSOPEN primitive. Then the command is sent through the medium of the Network Exchange Handler. The Remote Administrative Exchange Handler (RAEH server) must be started when submitting a command to a remote system.

The optional STATION parameter is used to define the name of the station specified in the command, if any. It modifies the semantics of the command.

The optional EVERY parameter is used to cause a repetition of the command at the interval given in seconds by the value of the parameter. A response to a repeatable command, which is not being delivered, is refreshed by the following response as soon as it is available. The requestor is thus assured of receiving the latest available response. The repetition of a command stops at the termination of the connection, in the case of an explicit request by H_PMSCLOSE, or in the case of normal or abnormal task termination by an implicit request. Finally, the number of outstanding commands is restricted to a certain value (32 in the current release). In the same way, the space of the various chains of command/responses in the DOF 7-PO queue file is allocated according to a quota so that the system tables cannot overflow through overuse by a single user. In the case of overflow, any submission of a command is refused until enough space is available.

6. The C Language Functions

This section describes the DOF 7-PO programming interface in the C language.

	C interface	GPL macro
kernel descriptor	struct _pms_kdesc	H_DCPMSKDESC
semaphore-messages	struct _pms_cmdsemmsg struct _pms_lnksemmsg struct _pms_msgsemmsg	H_DCPMSSEMMSG
parameters	struct _pms_param	-
message queue status	struct _pms_qstatus	-
semaphore characteristics	struct _pms_sem	-
user identification	struct _pms_userid	-
close DOF 7-PO connection	h_pmsclose	H_PMSCLOSE
convert site id.	h_pmscvsite	H_PMSCVSITE
edit message	h_pmsedtmsg	H_PMSEDTMSG
get message	h_pmsgetmsg	H_PMSGETMSG
get message first record	h_pmsgetmsgfr	H_PMSGETMSGFR
get response	h_pmsgetrp	H_PMSGETRP
get response first record	h_pmsgetrpfr	H_PMSGETRPFR
Initialize connection	h_pmsopen h_pmsopen_userid	H_PMSOPEN
send command	h_pmssendcmd	H_PMSSENDCMD

Notes for C Language Programmers

The following notes apply when using the DOF 7-PO interface in C language.

- It is extremely important for char[n] arrays that many C data structures map directly into the equivalent GPL structures. Some char[n] arrays are not C strings, but are left justified byte arrays with space padding to the right. These arrays do not require a C NULL character at the end.

- The GCOS 7 native language does not support zero as the value of the NULL pointer. Instead, it supports `0xffffffffL`. For this reason, some of the header files, dedicated to GCOS 7 system services, contain the following declaration:

  ```
  const char *NULL_PTR = 0xffffffffL;
  ```

  Because the assignment of an arithmetic value to a pointer is not portable, warnings would be issued from the compilation when those header files are included.

- Some of the interfaces are implemented as functions. Each one is documented with the returned type. For example, the following returns an integer value:

  ```
  int h_testrc(rc)
  unsigned long rc;
  ```

- You must check the return code that the system primitives give. To do this, use the `h_testrc` function, shown in the following example:

  ```
  {h_putjor (message, length)}
  if (!h_testrc (DONE))
  {/* The h_putjor primitive is not successfully completed*/}
  else {/* The h_putjor primitive is successfully completed*/}
  ```

- You can also check the class of the return code: Normal or Abnormal. This is shown in the following example:

  ```
  {h_jobtime (jobtime, step_limits, ron);}
  if (h_testrc (ABNORMAL))
  {/*For the primitive h_jobtime, the Abnormal Return-Code is:*/
  /*                                          NOMATCH or      */
  /*                                          JNBERR or       */
  /*                                          SWAPPED         */
  }
  { /* The Return-Code is DONE.                               */
  }
  ```

- Some C language interface primitives are replaced, after preprocessing, by a sequence of declarations, followed by a sequence of statements. These macros must be enclosed between braces by the user and are documented as follows:

  ```
  { h_jobident (job,option) }
  struct _jobident job;       /* output parameter */
  enum {SHORT,DETAILED} option;
  ```

- Arguments are input parameters unless otherwise specified inside the comments in the function declaration description. This is the case in the above example, where `job` specifies the name of a structure that the system service fills and `option` is an input parameter.

- Any C language program calling the interfaces described in this document must be compiled with the LEVEL=GCOS 7 option. This feature provides the call-by-address parameter passing.
- All the system header files are members of the SYS.C.INCLUDE library.

For the C language interface, the inclusion header file is <pmos.h>.

The pmos.h Header File

The <pmos.h> file contains definitions that are used as arguments of some functions:

```
const char *NULL_PTR=0xffffffffL;

#define PMS_KDESCLN 47

#define PMS_LNKRI 4095

#define PMS_FMT_GCOS "GCOS"
#define PMS_FMT_AEP  "AEP "

#define PMS_LOCAL_SITE   "\377\377\377\377"
#define PMS_DEFAULT_SITE "\377\377\377\377"
#define PMS_LOCAL_STATION "\377\377\377\377\377\377\377\377"

#define PMS_STILLMSG 1
#define PMS_DONE 2
#define PMS_EMPTY 4
#define PMS_LOST_PREVIOUS 1
#define PMS_LOST_FOLLOW 2
#define PMS_LOST 3

#define PMS_CLEAN      2
#define PMS_REPEATMSG  4
#define PMS_WAIT       8
#define PMS_FORCE     16

const struct _pms_param _pms_param_init = {
     0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0
};
```

NOTE:
The use of <pmos.h> would lead to five warnings due to non-portable initializations of pointers:

```
const char *NULL_PTR = 0xffffffffL;
const struct _pms_param _pms_param_init = {
     0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0
};
```

6.1 Kernel Descriptor (STRUCT_PMS_KDESC)

Function

This structure maps the part of a descriptor that is common to all format managers.

Format

```
#include <pmos.h>
struct _pms_kdesc {
     char fmname[4];
     short descln, recln;
     char recname[8];
     struct {
          unsigned rectype:2, umtype:2, main:1, ckfm:1;
          unsigned bridge:1, nolog:1, mbz:8;
     } nature;
     char mbz[6], format[4], version;
     struct {
          char nbfield, maskln;
          short zoneln, tabln;
     } selection;
     struct {
          char nbfield, maskln;
          short zoneln, tabln;
     } modification;
     struct {
          char nbfield, maskln;
          short zoneln, tabln;
     } response;
};
```

Description of the structure

`fmname`	Name of the format manager ("OMH ", "DSAC",...). The string is not null terminated but right padded with blanks.
`descln`	Whole length of the descriptor.
`recln`	Whole length of the related record.
`recname`	Name of the related record. The string is not null terminated but right padded with blanks.
`nature`	Structure describing the nature of the related record.

`nature.rectype`	– unsolicited message. – command. – response.
`nature.umtype`	significant only for an unsolicited message. – informational message. – action message to be cancelled by the issuer. – action message to be acknowledged by the recipient. – question.
`nature.main`	significant only for an unsolicited message. – specific message. – generic message.
`nature.ckfm`	significant only for the OMH format manager. – record syntax not checked. – record syntax checked and correct.
`nature.bridge`	significant only for the OMH format manager. – record sent by means of the DOF 7-PO interface. – record sent by the OMH bridge.
`nature.nolog`	– the record may be logged in the SYS.LOGC file if necessary. – the record may not be logged in the SYS.LOGC file.
`format`	Name of the parameter format depending on a given format manager ("GCOS", "AEP", "EDIT",...). The string is not null terminated but right padded with blanks.
`version`	Version number of the descriptor.
`selection`	Description of the parameters of the selection region.
`modification`	Description of the parameters of the modification region.
`response`	Description of the parameters of the response region.
`region.nbfield`	Number of elementary fields contained in the region. For a response region, it may also include an iterative item that is not constant.

`region.maskln`	Length of the region presence mask. For a response region, the part of the presence mask related to the items is common to all items.
`region.zoneln`	Length of the area containing the elementary fields of the region. The absent fields are not significant. For a response region, it may include one variable item.
`region.tabln`	Effective length of the whole tables describing the elementary fields of the region.

6.2 "Response Available" Semaphore-Message (STRUCT_PMS_CMDSEMMSG)

Function

This structure maps the semaphore-message related to a DOF 7-PO event notified on the DOF 7-PO connection semaphore. This event stands for "Response available" and is related to the `cmdri` request identifier.

Format

```
#include <pmos.h>
struct _pms_cmdsemmsg {
     char c[6];            /* internal use only           */
     unsigned short cmdid; /* command identifier          */
     unsigned long pmsid;  /* DOF 7-PO identifier         */
     unsigned unused : 4;
     unsigned sysri : 12;  /* system request identifier   */
     unsigned short cmdri; /* response request identifier */
};
```

Description of the structure

`cmdid`	Command identifier specified at the issuing of the command by the `h_pmssendcmd` function.
`pmsid`	DOF 7-PO connection identifier obtained at the connection time by the `h_pmsopen[_userid]` function.
`sysri`	Specifies the request identifier given by the system. In case of a "Response available" event, it is set equal to 0.
`cmdri`	Specifies the response request identifier. In case of a "Response available" event, it takes the cmdri value (field of `_pms_sem structure`) specified at connection time by `h_pmsopen[_userid]`.

6.3 "DOF 7-PO (Un)Available" Semaphore-Message (STRUCT_PMS_LNKSEMMSG)

Function

This structure maps the semaphore-message related to a DOF 7-PO event notified on the DOF 7-PO connection semaphore. This event stands for "DOF 7-PO facilities (un)available" and is related to the execution of the `h_pmsopen[_userid]` function.

Format

```
#include <pmos.h>
struct _pms_lnksemmsg {
     char c[6];           /* internal use only         */
     unsigned short rc;   /* reason code               */
     unsigned long pmsid; /* DOF 7-PO identifier       */
     unsigned unused : 4;
     unsigned sysri : 12; /* system request identifier */
     unsigned short site; /* site identifier           */
};
```

Description of the structure

`rc` — Specifies the reason code related to the execution of `h_pmsopen[_userid],` as returned by the underlying layers (right part of the GCOS 7 return code).

`pmsid` — DOF 7-PO connection identifier obtained at connection time by the `h_pmsopen[_userid]` function.

`sysri` — Specifies the request identifier given by the system. In case of a "DOF 7-PO facilities (un)available" event, it is set equal to `PMS_LNKRI` (constant defined in the `<pmos.h>` header file).

`site` — Defines an internal identifier associated to the system specified at the initiation of the connection. To convert this identification into the corresponding external site name, the user's program may use the `h_cvsite` function. A null identifier corresponds by convention to the local system.

6.4 "Unsolicited Message Available" Semaphore-Message (STRUCT_PMS_MSGSEMMSG)

Function

This structure maps the semaphore-message related to a DOF 7-PO event notified on the DOF 7-PO connection semaphore. This event stands for "Unsolicited message available" and is related to the `msgri` request identifier.

Format

```
#include <pmos.h>
struct _pms_msgsemmsg {
     char c[6];            /* internal use only           */
     unsigned short flsid; /* filter set identifier       */
     unsigned long pmsid;  /* DOF 7-PO identifier         */
     unsigned unused : 4;
     unsigned sysri : 12;  /* system request identifier   */
     unsigned short msgri; /* message request identifier  */
};
```

Description of the structure

`flsid`	Filter set identifier giving access to a class of messages. For generic messages, this identifier is specified at the creation of a filter set by the CRFLTST command issued by the `h_pmssendcmd` function. For specific messages, this identifier has by convention the 0x4040 value.
`pmsid`	DOF 7-PO connection identifier obtained at the connection time by the `h_pmsopen[_userid]` function.
`sysri`	Specifies the request identifier given by the system. In case of a "Unsolicited message available" event, it is set equal to 0.
`msgri`	Specifies the message request identifier. In case of a "Unsolicited message available" event, it takes the msgri value (field of `_pms_sem structure`) specified at the DOF 7-PO connection time by `h_pmsopen[_userid]`.

6.5 DOF 7-PO Parameters (STRUCT_PMS_PARAM)

Function

The _pms_param type structures are used as input parameters of the following functions:

```
h_pmsedtmsg;
h_pmsgetmsg;
h_pmsgetrp;
h_pmssendcmd.
```

Format

```
#include <pmos.h>
struct _pms_param {
    char  *curdescptr;
    short curdescln, maxdescln;
    char  *descptr;
    short descln, maxrecln;
    short *recptr;
    short recln, maxfmparln;
    char  *fmparptr;
    short fmparln;
};
```

Comments

For a complete explanation of the structure parameters, refer to the functions using them. The following table shows the structure fields to be filled versus the DOF 7-PO application.

Elementary Field	edtmsg	Getmsg	Getrp	sendcmd
Curdescptr	-	N	N	-
Curdescln	-	O	O	-
Maxdescln	-	M	M	-
Descptr	M	M	M	M
Descln	M	O	O	M
Maxrecln	-	M	M	-
Recptr	N	M	M	N
Recln	N	O	O	N
Maxfmparln	-	N	N	-
Fmparptr	N	N	N	-
Fmparln	N	O	O	-

Input parameters:

M: mandatory;

N: mandatory. Must be null when meaningless (i.e. NULL_PTR for a pointer, 0 for an integer);

-: meaningless, not to be filled.

Output parameters:

O: filled by the called system service.

When the same parameter of `_pms_param` structure type is used more than once; the user should reset the structure before each use. It can be reset with the `_pms_param_init` constant variable that is defined in `<pmos.h>` as follows:

```
const struct _pms_param _pms_param_init = {
     0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0,0,0xffffffffL,0
};
```

6.6 Message Queue Status (STRUCT_PMS_QSTATUS)

Function

A `_pms_qstatus` type structure is used as an output parameter of either the `h_pmsgetmsg` or the `h_pmsgetrp` function. It gives the state of the piece of information delivered from the DOF 7-PO message queue. It specifies also whether some elements of the chain of unsolicited message or responses have been lost.

Format

```
#include <pmos.h>
struct _pms_qstatus {
    unsigned abnormal : 1;  /* abnormal termination           */
    unsigned unused : 4;
    unsigned delivered : 3; /* delivery state                 */
    char lost;              /* losing of queue elements       */
};
```

Description of the structure

`abnormal`
= 1: Abnormal termination.
= 0: Normal termination.

`delivered`
= PMS_STILLMSG: To be delivered.
= PMS_DONE: Being delivered. In this case, the return code of the function is usually DONE, however DONEIDE or WALIM may override it.
= PMS_EMPTY: Delivered. In this case, the return code of the function is usually DATALIM, however DONEIDE or WALIM may override it.

`lost`
= 0: No element lost.
= PMS_LOST_PREVIOUS: The first element of the chain of unsolicited messages or responses is delivered but some previous elements have been lost.
= PMS_LOST_FOLLOW: The current element is delivered but some following elements have been lost.
= PMS_LOST: The first element of the chain of unsolicited messages or responses is delivered but some previous and some following elements have been lost.

6.7 Semaphore Characteristics Structure (STRUCT_PMS_SEM)

Function

A _pms_sem type structure is an input parameter of the h_pmsopen[_userid] function. It contains the characteristics of the semaphore for events during a DOF 7-PO connection.

Format

```
#include <pmos.h>

struct _pms_sem {
    char c;              /* internal use only              */
    char *sem;           /* pointer to semaphore descriptor */
    unsigned short s1;   /* internal use only              */
    unsigned short msgri;/* message request identifier     */
    short msgprty;       /* message enqueueing priority    */
    unsigned short s2;   /* internal use only              */
    unsigned short cmdri;/* response request identifier    */
    short cmdprty;       /* response enqueueing priority   */
    unsigned long lnkri; /* internal use only              */
    short lnkprty;       /* event enqueueing priority      */
};
```

Description of the structure

sem	This field specifies the pointer to the descriptor of the semaphore. This semaphore is for the "Response available", "Unsolicited message available", and "DOF 7-PO facilities (un)available" events.
msgri	This field specifies the request identifier that the semaphore message returns at the time of the notification of the "Unsolicited message available" event.
msgprty	This field specifies the queue priority of the "Unsolicited message available" notification of the previous semaphore. Valid values are from 0 through 15, with 0 having the highest priority. This priority is valid for single or multiple connections. It defines a hierarchy among various events notified for the same semaphore. If the given priority is outside the two bounds, this priority is forced to 15.

`cmdri`	This field specifies the request identifier that the semaphore message returns at the time of the notification of the "Response available" event.
`cmdprty`	This field specifies the queue priority of the "Response available" notification of the previous semaphore. Valid values are from 0 through 15, with 0 having the highest priority. This priority is valid for single or multiple connections. It defines a hierarchy among various events notified for the same semaphore. If the given priority is outside the two bounds, this priority is forced to 15.
`lnkprty`	This field specifies the queue priority of the "DOF 7-PO facilities (un)available" notification of the previous semaphore. Valid values are from 0 through 15, with 0 having the highest priority. This priority is valid for single or multiple connections. It defines a hierarchy among various events notified for the same semaphore. If the given priority is outside the two bounds, this priority is forced to 15.

6.8 User Identification (STRUCT_PMS_USERID)

Function

A `_pms_userid` type structure is an input parameter of the `h_pmsopen_userid` function. It contains the identity and related station name of the requestor of the DOF 7-PO connection.

Format

```
#include <pmos.h>

struct _pms_userid {
    char c;              /* internal use only              */
    char usernm[12],password[12],project[12],billing[12];
    char station[8];
};
```

Description of the structure

`usernm`	This field specifies the identity of the DOF 7-PO requestor. This value is checked against the SITE.CATALOG. It must be right padded with blanks and have no terminating `\0`.
`password`	This field specifies the password related to the user name and is checked against the SITE.CATALOG. It must be right padded with blanks and have no terminating `\0`. If the catalog has no password, this field must contain only blanks.
`project`	This field specifies the user project. It must be right padded with blanks and have no terminating `\0`.
`billing`	This field specifies the billing related to the project. It must be right padded with blanks and have no terminating `\0`.
`station`	This field specifies the implicit station that relates to the current user's project. This station routes the outputs of the jobs submitted during the connection. It must be right padded with blanks and have no terminating `\0`.

6.9 Close (H_PMSCLOSE)

Function

This function terminates a DOF 7-PO connection.

Format

```
#include <pmos.h>

{ h_pmsclose(pmsid,site) }
  unsigned long pmsid;
  char *site;          /* or char site[5]; */
```

Description of the structure

`pmsid`	This parameter is the DOF 7-PO connection identifier obtained at the initiation of the connection.
`site`	Defines the target system for a remote termination. This string must terminate with a null character and be less than 5 characters long. To terminate the local connection, set `site` to `PMS_LOCAL_SITE` (constant string defined in the <`pmos.h`> header file).

Return codes

Normal

`DONE`	Connection termination completed normally.

Abnormal

`ARGERR`	There is an error in a parameter.
`NOTOPEN`	This connection is not initiated.
`OPTERR`	A remote system is specified to terminate a connection without access to the network.
`SEQERR`	The caller does not have the right to manage this connection.

Comments

The use of this function allows an application to interrupt any activity on the specified DOF 7-PO connection. The DOF 7-PO requestor or the system at process termination can implicitly request the termination of a connection.

The commands and the related responses that the requestor has not yet obtained are deleted from the DOF 7-PO queue file at the termination of the connection. Repetition of commands submitted with the "every" argument is ended.

The DOF 7-PO queue manager file keeps specific information messages. It stores new specific information messages that are sent during the termination of the connection of their recipient and delivers them at the next initiation of the concerned connection.

The generic information messages that are not yet received are deleted from the queue file. Also, the filter sets are deleted so that no generic messages can be stored during a termination of a connection.

The repeatable messages that the requestor receives are deleted from the DOF 7-PO queue file. The requestor, which belongs to the generic class of main operators, is removed from the list of recipients in the Repeatable Messages Table. The list is updated at the next repetition and the unsolicited message can be sent to other recipients.

The issuer of a specific question is either aborted or notified with a null reply according to the value of the NABORT parameter given when the question was issued.

The `site` parameter performs a selective termination on the remote connection or connection relay that the remote target system initiates. Using `h_pmsopen` with a non local site parameter initiates two DOF 7-PO connections: a local connection on the local system and a remote connection on the default target system specified by the site parameter. In the same way, when `h_pmssendcmd` is used with a site parameter that is different from the default target system (specified at the initiation of the connection), a remote connection is implicitly initiated on the remote target system.

A selective termination has no effect on the local connection. However, if the system given as a parameter is the system specified at the initiation of the DOF 7-PO connection, the parameter is ignored. The termination takes effect, in this case, on the local connection as well as on all the remote connection-relays. (See also the diagram under H_PMSCLOSE in Section 5.)

The following table summarizes the results of a termination with `h_pmsclose`.

Nature of the data	Treatment of the data
commands/responses	Deleted
specific information messages	Kept
generic information messages	Deleted
repeatable messages	Deleted

6.10 Convert Site Identifier (H_PMSCVSITE)

Function

This function converts the site internal identification into the corresponding external site name and vice versa.

Format

```
#include <pmos.h>

{ h_pmscvsite (input,ident,name) }
 enum {NAME,IDENT} input;
 unsigned short ident; /*input-output parameter */
 char name[5];         /*input-output parameter */
```

Description of the structure

`input`	This parameter gives the input format of the conversion:
`NAME`:	the site name is given as a character string;
`IDENT`:	the identification is given as a 16-bit string.
`ident`	The internal identification of the site. When input specifies `IDENT`, it is given as an input parameter. When input specifies `NAME`, it is given as an output parameter.
name	The null terminated string that specifies the external identification of the site. When input specifies NAME, it is given as an input parameter. When input specifies IDENT, it is given as an output parameter.

Return codes

Normal

DONE	Normal completion of the identification.

Abnormal

ARGERR	Unknown site internal identification. This return code indicates that input=IDENT.
NOINIT	The Remote Administrative Exchange Handler (RAEH) was not started.
RSUNKN	The site given in input is not been found in the RAEH tables. This return code indicates that input=NAME.

Comments

The correspondence between a site internal identification and an external site name is established only for the lifetime of the DOF 7-PO session on the remote site. This span is between the opening of the session on the remote site until its closing (h_pmsclose), or until the receipt of an abnormal termination event (PMS_LNKRI message).

Therefore, the successful conversion between a site internal identification and an external site name is guaranteed only for this period of time. After the DOF 7-PO session ending, a call to h_pmscvsite can be tried but with no guarantee of success.

6.11 Edit Message (H_PMSEDTMSG)

Function

To edit a command, a response or an unsolicited message contained in a structured record. In DOF 7-PO, this means to translate the contents of the record into a character string, replacing the variables with values.

Format

```
#include <pmos.h>

{ h_pmsedtmsg (param,message,maxln,MSG) }
 struct _pms_param param;
 struct {
      short length;
      char text[maxln+1];
 } message;     /* input-output parameter  */
int maxln;
```

Description of the structure

`param.descptr`	This parameter contains the pointer to the descriptor of the record to edit.
`param.descln`	This parameter gives the length of the descriptor.
`param.recptr`	This parameter contains the pointer to the record to edit. When there is no variable field in the related message, `NULL_PTR` must be specified.
`param.recln`	This parameter contains the length of the record to edit. When there is no variable field in the related message, zero must be specified.
`param.fmparptr`	This parameter contains the pointer to a parameter area furnished by the related format manager and obtained with the `h_pmsgetrp` or `h_pmsgetmsg` functions. This area contains control information useful for an edition. When meaningless, `NULL_PTR` must be specified.
`param.fmparln`	Length of the parameter area to which `param.fmparptr` points. When meaningless, zero must be specified.

`message`	Area to receive the edited record. The edited message is returned in `message.text` and terminates with `\0`. `Message.length` gives the output length and does not include the terminating null character.

`maxln`	The maximum length of the area declared for the edited message. `Maxln` does not include the terminating `\0` character. Truncation occurs if the output length exceeds `maxln` and the `TRUNC` return code is given back.
	Also, when a truncation occurs, the following message is returned:
	`message.text[maxln]='\0'`
	Otherwise, the following message is returned:
	`message.text[message.length]='\0'`

Return codes

Normal	
`DONE`	Function completed normally.
Abnormal	
`ARGERR`	Error in a parameter.
`CATERR`	The object that the catalog pointer points to does not have the characteristics of a catalog.
`DAMAGED`	Inconsistency in the binary description of a message.
`DESCERR`	Inconsistency related to the descriptor of the message.
`FLNAV`	The specified catalog is not available.
`NMTCHERR`	The presence mask or the fields of the record have a value that is not consistent with the descriptor.
`TPUNKN`	The message was not found in the catalog.
`TRUNC`	The length of the edited message exceeds `maxln`. This message is truncated to the value specified by `maxln`.

Comments

This function edits a cataloged command, response or unsolicited message according to specified variable parameters given by `param.recptr`. The command, response, or unsolicited message is then stored in the output area.

The content of the catalog that relates to the structure drives the edition of a cataloged command. If the structure contains variable parameters, then the value of these are put in the record pointed to by `param.recptr`.

For system objects, the editing of a cataloged command, response, or unsolicited message is driven by the content of the catalog related to that structure. If the structure contains variable parameters, then the values of these are put in the REC record. If the catalog contains a 'new line' indicator, the two characters carriage return and line feed ("\015\045") are inserted in the output area to indicate a new line.

For network objects, the length of output lines is defined by the OUTPUT_LENGTH field of the MESSAGE parameter. The two characters carriage return and line feed ("\015\045") are inserted in the output area at the end of each line to indicate a new line.

6.12 Get Message (H_PMSGETMSG)

Function

This function allows the program to get an unsolicited message belonging to a class of messages defined by a filter set.

Format

```
#include <pmos.h>

{ h_pmsgetmsg (pmsid,flsid,msgid,param,nextrecname,state) }
 unsigned long pmsid;
 unsigned short flsid;
 unsigned short msgid;      /* output parameter       */
 struct _pms_param param;   /* input-output parameter */
 char nextrecname[8];       /* output parameter       */
struct _pms_qstatus state;  /* output parameter       */
```

Description of the structure

`pmsid`	This parameter identifies the DOF 7-PO connection.
`flsid`	This parameter identifies the filter set that access an unsolicited message.
`msgid`	This parameter receives an unsolicited message identifier when the delivered unsolicited message is repeatable. The descriptor of the unsolicited message contains the type of the unsolicited message (for example, action or question). When the unsolicited message is an information message, this parameter takes the conventional 0xffff value.
`param.maxdescln`	Maximum size of the area that receives the descriptor that describes the unsolicited message. If the descriptor length exceeds this size, truncation occurs and a `WALIM` return code is given back.
`param.maxrecln`	Maximum size of the area that receives the unsolicited message. If the record length exceeds this size, truncation occurs and a `WALIM` return code is given back.

`param.descptr`	This is the pointer to the descriptor describing the returned unsolicited message.
`param.descln`	This is the length of the returned descriptor.
`param.recptr`	This is the pointer to the record of the returned unsolicited message.
`param.recln`	Length of the returned record.
`param.maxfmparln`	Maximum size of the area that receives a structure that the format manager builds, which is related to the delivered response. If the structure length exceeds this size, truncation occurs and a `WALIM` return code is given back.
`param.fmparptr`	This parameter points to a parameter area that the format manager builds, which is related to the delivered unsolicited message. This area contains control information useful for an edition.

`param.fmparln`	This is the current length of the parameter area pointed to by `param.fmparptr`. When `param.fmparptr` is set to `NULL_PTR`, it is meaningless and zero is returned.
	To not receive any complementary structure that the related format manager builds, specify the following:
	`param.fmparptr=NULL_PTR;`
	`param.maxfmparln=0;`
	`param.curdescptr`
	
	This points to the descriptor of the expected structured record of the current unsolicited message. If this descriptor is different from the one sent, the related format manager can do a translation. To not use this facility, specify `NULL_PTR`.

`nextrecname`	This is the name of the next delivered unsolicited message. This parameter allows the application to give a pointer at the next call of `h_pmsgetmsg` to the descriptor of the expected structured record in this unsolicited message. The specified array is returned, right padded with blanks. There is no terminating `\0` character. When `the param.curdescptr` facility is not used, `nextrecname` is ignored and left unchanged.

`state`	This parameter gives the state of the message queue. It also gives a return code upon the execution of `h_pmsgetmsg`.

Return codes

Normal

`ALMOST`	Function completed normally: other items of the same unsolicited message remain to be delivered. This feature is not used by the current format managers.
`DATALIM`	Function completed normally: the last unsolicited message is fully delivered. This class of messages is now empty. In this case, the state.delivered field takes the value `PMS_EMPTY`.
`DONE`	Function completed normally: the current unsolicited message is fully delivered. It remains some unsolicited messages belonging to this class. In this case, the state.delivered field takes the value `PMS_DONE`.
`DONEIDE`	Unsolicited message delivered, but the translation required by giving the `param.curdescptr` parameter cannot be done. The response is delivered as it was sent.
`WALIM`	The unsolicited message is delivered but truncation has occurred because the output length exceeds `param.maxdescln`, `param.maxrecln` or `param.maxfmparln`.

Abnormal

`ARGERR`	Error in a parameter.
`DESCERR`	Incoherence related to the descriptor of the unsolicited message.
`EXHAUST`	The last unsolicited message of this class has already been delivered.
`INDUNKN`	Incorrect value of the filter set identifier.

ITMNAV	The DOF 7-PO requestor has not been notified of the availability of an unsolicited message related to this class.
NOTOPEN	This connection was not initiated.
SEQERR	The caller does not have the right to manage this connection.
TOOLATE	The previously available unsolicited message is no longer significant because it is a repeatable message that another recipient already takes into account. Because of this, there is no more unsolicited message of this class available.

Comments

This function obtains one at a time, the unsolicited messages sent to a DOF 7-PO requestor.

The identifier of the class of the required unsolicited messages is given in the `flsid` parameter. The blank identifier (0x4040) is the conventional identifier for specific messages. The name of the filter set identifies the class of generic messages.

The filters are created by the CRFLTST and CRFLT commands, which can be submitted from an application by means of the `h_pmssendcmd` function. The name of a filter set defines the name of a class of messages. A specific message is therefore always sent to its recipients, whether the connection is initiated or not, while a generic message is sent to the DOF 7-PO requestor(s) with filters matching the unsolicited message.

The semaphore message related to the "Unsolicited message available" event is declared with the `_pms_msgsemmsg` structure type:

```
struct _pms_msgsemmsg {
     char c[6];            /* internal use only           */
     unsigned short flsid; /* filter set identifier       */
     unsigned long pmsid;  /* DOF 7-PO identifier         */
     unsigned unused : 4;
     unsigned sysri : 12;  /* system request identifier   */
     unsigned short msgri; /* message request identifier  */
};
```

`flsid` is the filter set identifier giving access to that class of messages and `pmsid` is the DOF 7-PO connection identifier. The `sysri` field is 0. `Msgri` is the request identifier specified at the initiation of the connection and related to the "Unsolicited message available" event.

The notification on the semaphore given at the initiation of the connection is performed only after the first creation of a filter set, or each time all unsolicited messages have been received by an application. This is when an unsolicited message is stored in an empty class of messages. The state of the class of messages then takes the value PMS_STILLMSG (to be delivered).

When the `h_pmsgetmsg` function is called, the system returns an unsolicited message corresponding to the specified identifier (the `state.delivered` field takes the value PMS_DONE, that is, being delivered) and deletes the unsolicited message from the DOF 7-PO queue file, so that the next call returns the next unsolicited message of the same class. When the class becomes empty, a special return code is given back with the last unsolicited message (the `state.delivered` field takes the value PMS_EMPTY, that is, delivered) and the next issuing of an unsolicited message belonging to this class results in a notification on the connection semaphore.

The unsolicited message, or the descriptor and the related record, is given back by default to the requestor as it was sent by the issuer. However, the requestor can know the name of the next record, before receiving it. The name of the first record is given by the `h_pmsgetmsgfr` function. Each `h_pmsgetmsg` function gives the name of the next record. This name is meaningless for the last unsolicited message of the class, and it is set to "space". The `param.curdescptr` parameter specifies the descriptor describing the structured record of the unreceived unsolicited message. The object manager and the DOF 7-PO requestor may have different version numbers. The access method ensures compatibility in the exchanged structures so that the two DOF 7-PO service users understand each other.

The DOF 7-PO applications may receive, optionally, a complementary structure built by the related format manager. This structure contains the filtering criteria in the case of the OMH format manager, or editing indicators or status information for the DSAC format manager. We recommend that you specify this structure when using DSAC.

With two exceptions, the output areas are not modified if the function is not successfully completed. The exceptions are as follows:

- `param.curdescln`, which returns the size of the expected record descriptor.
- `param.fmparln`, which returns 0 if `param.fmparptr` is set equal to `NULL_PTR`.

6.13 Get Message First Record Name (H_PMSGETMSGFR)

Function

This function gets the record name of the first unsolicited message of the specified class.

Format

```
#include <pmos.h>

{ h_pmsgetmsgfr (pmsid,flsid,format,recname) }
 unsigned long pmsid;
 unsigned short flsid;
 char *format;
char recname[8];       /* output parameter */
```

Description of the structure

`pmsid`	This parameter identifies the DOF 7-PO connection.
`flsid`	This parameter identifies the filter set that gives access to an unsolicited message.
`format`	This parameter contains the format of the record related to the unsolicited message. Set this equal to `PMS_FMT_GCOS` or `GCOS_FMT_AEP`, which are constants defined in the `<pmos.h>` header file.
`recname`	This parameter is the name of the record related to the next delivered unsolicited message. This next message corresponds to the named filter set. The specified string is returned right padded with blanks up to 8 characters. There is no terminating `\0` character.

Return codes

Normal

`DONE`	Function completed normally.

Abnormal

ARGERR	There is an error in a parameter.
EXHAUST	The last unsolicited message of this class has already been delivered.
INDUNKN	The filter set identifier has an incorrect value.
ITMNAV	The DOF 7-PO requestor is not notified of the availability of an unsolicited message related to this class.
NOTOPEN	This DOF 7-PO connection is not initiated.
SEQERR	The caller does not have the right to manage this connection.
TOOLATE	The previously available unsolicited message is no longer significant. This is because it is a repeatable message that another recipient has already taken into account. When this return code occurs, no more unsolicited messages of this class are available.

Comments

This function allows the requestor to get the record name of the first unsolicited message related to a class of messages. The `pmsid` and `flsid` pair specifies the class.

The semaphore message gives this pair at the notification time of the "Unsolicited message available" event.

The semaphore message related to the unsolicited-message-available event is declared with the `following _pms_msgsemmsg` structure type:

```
struct _pms_msgsemmsg {
char c[6];            /* internal use only          */
unsigned short flsid; /* filter set identifier      */
unsigned long pmsid;  /* DOF 7-PO identifier        */
unsigned unused : 4;
unsigned sysri : 12;  /* system request identifier  */
unsigned short msgri; /* message request identifier */
};
```

`flsid` is the filter set identifier that gives access to that class of messages, and `pmsid` is the DOF 7-PO connection identifier. Set the `sysri` field equal to 0. The initiation of the connection specifies `msgri`, which is related to the Unsolicited message available event.

For an unsolicited message related to the DSAC format manager, the expected format of the records is GCOS or AEP. If the expected format is the GCOS format, then the `recname` field is filled with the CLASS abbreviation and the CODE abbreviation. The rest of the field is padded with spaces. If the expected format is the AEP format, the recname field is the CLASS number and the CODE number, with the rest of the field padded with binary zeroes.

For an unsolicited message related to the OMH format manager, the expected format of the records is GCOS.

6.14 Get Response (H_PMSGETRP)

Function

To get a response, or an element of a response, to a command which is defined by its identifier.

Format

```
#include <pmos.h>

{ h_pmsgetrp (pmsid,cmdid,param,nextrecname,state)}
 unsigned long pmsid;
 unsigned short cmdid;
 struct _pms_param param;     /* input-output parameter */
 char nextrecname[8];         /* output parameter       */
 struct _pms_qstatus state;   /* output parameter       */
```

Description of the structure

`pmsid`	This parameter identifies the DOF 7-PO connection concerned.
`cmdid`	This parameter identifies the command.
`param.maxdescln`	This parameter is the maximum size of the area to receive the descriptor describing the response. If the descriptor length exceeds this size, truncation occurs with a `WALIM` return code.
`param.maxrecln`	This parameter is the maximum size of the area to receive the response. If the record length exceeds this size, truncation occurs with a `WALIM` return code.
`param.descptr`	This parameter points to the descriptor describing the returned response.
`param.descln`	This parameter is the length of the returned descriptor.
`param.recptr`	This parameter points to the record of the returned response.
`param.recln`	This parameter is the length of the returned record.

`param.maxfmparln`	This parameter is the maximum size of the area to receive a structure that the format manager builds. This is related to the delivered response. If the structure length exceeds this size, truncation occurs with a `WALIM` return code.
`param.fmparptr`	This is the pointer to a parameter area that the format manager builds. This is related to the delivered response. This area contains control information useful for an edition.

`param.fmparln`	This parameter is the current length of the parameter area pointed to by `param.fmparptr`. When `param.fmparptr` is meaningless, zero is returned. It is meaningless when the user sets it to `NULL_PTR`. To not receive any complementary structure built by the related format manager, the following must be specified:
	
	`param.fmparptr=NULL_PTR;`
	`param.maxfmparln=0;`

`param.curdescptr`	This parameter points to the descriptor of the expected structured record of the current response. If this descriptor is different from the one sent, the related format manager can do a translation. To not use this facility, specify `NULL_PTR`.
`nextrecname`	This parameter names the next delivered response. At the next call of `h_pmsgetrp`, this parameter allows the application to give a pointer to the descriptor of the expected structured record of this response. The specified array is returned right padded with blanks. There is no terminating `\0` character. When `param.curdescptr` facility is not used, `nextrecname` is ignored and left unchanged.
`state`	This parameter gives the state of the message queue. It also gives a return code after the execution of `h_pmsgetrp`.

Return codes

Normal

ALMOST	The function completed normally. Other items of the same response still remain. The current format managers do not use this feature.
DATALIM	The function completed normally, and the last response is fully delivered. No responses remain related to this command. The state.delivered field takes the value PMS_EMPTY.
DONE	The function completed normally. The current response is fully delivered and some responses related to this command remain. The state.delivered field takes the value PMS_DONE.
DONEIDE	Response delivered, but the translation required by giving the param.curdescptr parameter cannot be done. The response is delivered as it was sent.
WALIM	The response is delivered but truncation occurs because the output length exceeds param.maxdescln, param.maxrecln or param.maxfmparln.

Abnormal

ARGERR	There is an error in a parameter.
DESCERR	There is incoherence related to the descriptor of the response.
EXHAUST	The last response related to this command is already delivered.
INDUNKN	Incorrect value of the command identifier.
ITMNAV	The response is not available. The semaphore of the DOF 7-PO connection is not notified.
NOTOPEN	This DOF 7-PO connection is not initiated.
SEQERR	The caller does not have the right to manage this connection.
SYSOV	The response is not complete because of space overflow in the file used by DOF 7-PO to store the chains of commands and their related responses.

Comments

This function obtains the elementary responses (one at a time) that are related to a given command specified by the command identifier.

Every time an object manager sends a response to a given command, this response is chained to the command in the DOF 7-PO queue file. When the last response is sent, the submitter of the command is notified on the semaphore given at the initiation of the connection.

The semaphore message related to the response available event is declared with the following `_pms_cmdsemmsg` structure type:

```
struct _pms_cmdsemmsg {
     char c[6];            /* internal use only            */
     unsigned short cmdid; /* command identifier           */
     unsigned long pmsid;  /* DOF 7-PO identifier          */
     unsigned unused : 4;
     unsigned sysri : 12;  /* system request identifier    */
     unsigned short cmdri; /* response request identifier  */
};
```

`cmdid` is the command identifier and `pmsid` is the DOF 7-PO connection identifier. The `sysri` field should be equal to 0. `cmdri` is the request identifier specified at the initiation of the connection and related to the response available event.

The notification on the semaphore given at the initiation of the connection is performed only when the last response has been stored in the DOF 7-PO queue file. When `h_pmsgetrp` is called, the system returns the current elementary response corresponding to the specified identifier, and the queue pointer to the response to be delivered is updated so that the next call returns the next response of the same chain. When the last response is obtained, a special return code is given back with it, and the whole chain is deleted from the queue file (the whole chain is the command and all the related responses).

The response is given back by default to the DOF 7-PO requestor as it was sent by the issuer. The response is the descriptor and the related record. However the requestor can have the name of the next record to be received before getting the response. First, the `h_pmsgetrpfr` field gives the name of the first record. Then each call of the `h_pmsgetrp` gives the name of the next record. This name is meaningless for the last response of the chain, it is set to "space". The DOF 7-PO requestor uses the param.curdescptr field to specify the descriptor describing the structured record of the response. The object manager and the DOF 7-PO requestor may have different version numbers. The access method ensures compatibility in the exchanged structures so that the two DOF 7-PO service users understand each other.

When a response is not fully delivered, the `h_pmsgetrpfr` function and the `h_pmsgetrp` allow the requestor to obtain the first elementary response function. The `h_pmsgetrpfr` function moves the current pointer to the first response of the chain, and the `h_pmsgetrp` function receives the response.

The applications can receive optionally a complementary structure built by the related format manager. This complementary structure of parameters can be used at the edition time of the response if editing is to be done.

With two exceptions, the output areas are not modified if the function is not successfully completed. The exceptions are as follows:

- `param.curdescln`, which returns the size of the expected record descriptor.
- `param.fmparln`, which returns 0 if `param.fmparptr` is set equal to `NULL_PTR`.

6.15 Get Response First Record Name (H_PMSGETRPFR)

Function

This function gets the record name of the first response of the specified command.

Format

```
#include <pmos.h>

{ h_pmsgetrpfr (pmsid,cmdid,recname) }
 unsigned long pmsid;
 unsigned short cmdid;
 char *format;
char recname[8];       /* output parameter */
```

Description of the structure

`pmsid`	This parameter identifies the DOF 7-PO connection.
`cmdid`	This parameter identifies the command.
`format`	This parameter is the expected format of the record related to the response. It should be set equal to `PMS_FMT_GCOS` or `GCOS_FMT_AEP`, which are constants defined in the `<pmos.h>` header file.
`recname`	This parameter is the name of the record related to the first delivered response corresponding to the named command/responses chain. The specified string is returned right padded with blanks up to 8 characters. There is no terminating `\0` character.

Return codes

Normal

`DONE`	The function completed normally.

Abnormal

ARGERR	There is an error in a parameter.
INDUNKN	Incorrect value of the command identifier.
ITMNAV	The response is not available. The semaphore of the DOF 7-PO connection is not notified.
NOTOPEN	This DOF 7-PO connection is not initiated.
SEQERR	The caller does not have the right to manage this connection.
SYSOV	The response is not complete because of space overflow. The overflow is in the file used by DOF 7-PO to store the chains of commands and related responses.

Comments

This function allows the requestor to get the record name of the first response associated to a command specified by the `pmsid`, `cmdid` pair. This pair has been obtained in the semaphore message at the time of notification of the response available event.

The semaphore message related to the response available event is declared with the following `_pms_cmdsemmsg` structure type:

```
struct _pms_cmdsemmsg {
     char c[6];            /* internal use only           */
     unsigned short cmdid; /* command identifier          */
     unsigned long pmsid;  /* DOF 7-PO identifier         */
     unsigned unused : 4;
     unsigned sysri : 12;  /* system request identifier   */
     unsigned short cmdri; /* response request identifier */
};
```

`cmdid` is the command identifier and `pmsid` is the DOF 7-PO connection identifier. The `sysri` field should be equal to 0. `cmdri` is the request identifier specified at the initiation of the connection and related to the response available event.

When a response is not fully delivered, the DOF 7-PO requestor can obtain the first elementary response with the `h_pmsgetrpfr` function and the `h_pmsgetrp` function. The `h_pmsgetrpfr` function moves the current pointer to the first response of the chain, and the h_pmsgetrp function receives the response.

In the case of a response related to the DSAC format manager, the expected format of the records are GCOS or AEP. If the expected format is the GCOS format, then the `recname` field is filled with the CLASS abbreviation and the CODE abbreviation. The rest of the field is padded with spaces. If the expected format is the AEP format, then the `recname` field is filled with the CLASS number and the CODE number; the rest of the field is padded with binary zeroes.

In the case of an unsolicited message related to the OMH format manager, the expected format of the records is GCOS.

6.16 Open (H_PMSOPEN)

Function

This function initiates a DOF 7-PO connection. When using h_pmsopen, the connection requestor is the submittor. By using h_pmsopen_userid, the connection can be opened under another user name.

Format

```
#include <pmos.h>

{ h_pmsopen (pms_sem,pmsnb,option,site,pmsid) }
 struct _pms_sem pms_sem;
 int pmsnb;
 int option;
 char *site;            /* or char site[5]; */
 unsigned long pmsid;   /* output parameter */

{ h_pmsopen_userid (pms_sem,pmsnb,option,userid,site,pmsid) }
 struct _pms_sem pms_sem;
 int pmsnb;
 int option;
 struct _pms_userid userid;
 char *site;            /* or char site[5]; */
 unsigned long pmsid;   /* output parameter */
```

Description of the structure

pms_sem	This parameter specifies the _pms_sem type structure which contains the characteristics of the semaphore on which will be notified various events during a DOF 7-PO connection.
pmsnb	This parameter specifies the number of the connection. For a given requestor, pmsnb must be unique in the system. Valid values are from 0 through 15.
option	This parameter is any of the following added together:

PMS_CLEAN	Purges all unsolicited messages available for this requestor at the initiation of the connection. This purge applies only to specific messages. This is because the generic messages are purged at the previous termination of the connection (if any). When this option is not specified, the requestor is notified if unsolicited messages are available.
PMS_REPEATMSG	Allows the repetition of repeatable messages after the requestor has received, but not yet acknowledged or replied to them. If this option is not specified at the initiation of the connection, then the repeatable messages are sent to this requestor only once during the connection.
PMS_WAIT	Requests waiting on a semaphore in the case of unavailability of the DOF 7-PO kernel as long as DOF 7-PO is unavailable. The h_pmsopen[_userid] function ends when the kernel is available and the connection initiated. When DOF 7-PO is implemented as a server, it is unavailable before SYSTEM READY. However, RERUN can restart after a system crash of the applications and before the DOF 7-PO server. Therefore a h_pmsopen[_userid] function can be submitted before the availability of DOF 7-PO and this keyword allows the user to handle this case.
PMS_FORCE	Initiates a connection in the range of connections reserved for the remote requestors. The site parameter is ignored.

When none of the previous options is to be applied, option must be set equal to 0.

userid	This parameter specifies the _pms_userid type structure which contains the identity of the DOF 7-PO requestor and the name of the related station. The userid string fields must not be terminated with \0, but must be right padded with blanks.

site	This parameter initiates a DOF 7-PO connection on the specified system. This string is null terminated and less than 5 characters long. By default, all commands submitted during this connection will be sent to this specified system. To initiate a local connection, site must be set equal to PMS_LOCAL_SITE, which is a constant string defined in the <pmos.h> header file. When the option specifies PMS_FORCE, the site is ignored.
pmsid	This parameter identifies the DOF 7-PO connection given by the system. A user can simultaneously initiate several connections, so the pmsid specifies the DOF 7-PO connection for a DOF 7-PO function. If the initiation is not completed then the pmsid parameter is filled with the conventional value 0xffffffff.

Return codes

Normal	
DONE	Function completed normally.
Abnormal	
ARGERR	Error in a parameter.
ENTRYOV	No more entries available in the system tables. There are 16 requestors having the MAIN right that are already connected.
NOINIT	The DOF 7-PO kernel is unavailable.
PROJUNKN	The project is unknown because it is not found in catalog.
PSWVIOL	There is a password violation.
RLBPUNKN	The relation billing/project is unknown because it is not found in catalog.
RSUNKN	The remote system is unknown.
SHUTDOWN	GCOS 7 is in the shutdown phase so new connections are refused.
STTNUNKN	The station is unknown because it is not found in catalog.
USERUNKN	The user is unknown because it is not found in catalog.
WITHSYN	The pair of user name/pmsnb has a synonym in the system tables.

Comments

The application uses `h_pmsopen[_userid]` to identify itself to the system and to initiate a connection with the DOF 7-PO kernel. A DOF 7-P application is a batch or an interactive application. When using `h_pmsopen`, the connection is initiated for the submitter of the job. When using `h_pmsopen_userid`, it is initiated for the user whose name, project, billing and station are specified. Any user known to the system catalog can initiate a DOF 7-PO connection.

The system deduces from this identification the access rights to the system commands. The operator access rights related to the project of the command issuer must be compatible with the access rights required to issue the command.

No more than 16 requestors having the MAIN right can be connected at the same time.

The same user can simultaneously initiate sixteen DOF 7-PO connections by qualifying them with a DOF 7-PO connection number. The pair that contains the user name and the DOF 7-PO connection number must be unique in the system. It is this pair that completely identifies the DOF 7-PO requestor. For best results, specify a user name at connection time.

The system returns a DOF 7-PO identifier, which must be used later with all functions, related to this connection. 5000 DOF 7-PO connections may be initiated on a system at the same time.

Through the `pms_sem` parameter, the requestor specifies the address of a semaphore at the initiation of a connection. It is on this semaphore that the response available, unsolicited message available, or DOF 7-PO facilities (un)available events are notified.

The requestor specifies the request identifiers that will be given back in the semaphore message at the time of the notification of the response available, unsolicited message available or DOF 7-PO facilities (un)available events. It also specifies the priorities related to those notifications.

Some checks are performed when a connection is initiated. As these checks can last a long time, initiation is always asynchronous. When the local checks are done, the application receives a return code, which is `DONE` if the checks are successful, but command submission is not yet allowed. When the connection is fully completed, a notification is sent on the semaphore related to the DOF 7-PO connection.

The semaphore message related to the DOF 7-PO facilities (un)available event is declared with the _pms_lnksemmsg structure type, as follows:

```
struct _pms_lnksemmsg {

     char c[6];           /* internal use only         */
     unsigned short rc;   /* reason code               */
     unsigned long pmsid; /* DOF 7-PO identifier       */
     unsigned unused : 4;
     unsigned sysri : 12; /* system request identifier */
     unsigned short s;    /* internal use only         */
};
```

The rc field is the reason code of the execution and the pmsid field is the DOF 7-PO session identifier. The sysri field is the request identifier related to the DOF 7-PO facilities (un)available event. It should be equal to PMS_LNKRI.

If the DOF 7-PO kernel is not available when the h_pmsopen[_userid] function is called, then an abnormal NOINIT return code is given back except if the PMS_WAIT option has been used. In this case, the application is set in a wait state until the kernel is available again.

When the kernel becomes unavailable, outstanding DOF 7-PO requests are aborted with an abnormal return code and new requests are rejected. The application must in this case request a new DOF 7-PO connection by means of the h_pmsopen[_userid] function, possibly with the PMS_WAIT option.

Several quotas are allocated to each DOF 7-PO connection in order so that a single user cannot cause an overflow of the DOF 7-PO queue file. These quotas are defined by the maximum number of the outstanding objects handled during a connection and by the size of each object. The maximum number of outstanding commands is set to 32. The maximum number of outstanding filter sets is set to 16. The whole size of a command chain is set to 512K. The size of a command chain includes a chain of a command and its related responses. The whole size of a chain of unsolicited messages is set to 64K. The whole size of a message chain includes a chain of unsolicited messages related to a given filter set or is the chain of the specific messages.

To give an idea of the capacity of a chain of 64 Kbytes - a chain of this size can store 128 JB08 messages.

When the quota related to the number of commands is reached, any command submission is rejected until the number of outstanding commands goes below the maximum allowed. When the quota related to the size of a handled chain is reached, any issuing of response or unsolicited message is rejected until the application clears the chain.

The PMS_CLEAN option is used to purge the chain of specific messages at the initiation of the connection. If the PMS_CLEAN option is not used, and this is the default mode, the application is notified on its connection semaphore if unsolicited messages are available.

The PMS_REPEATMSG option allows an application to receive the repeatable messages at each repetition. By default, the application receives this unsolicited message only once.

The site parameter defines the system on which all commands are submitted by default. If this parameter specifies PMS_LOCAL_SITE, the system is the local one. If the specified system is a DPS 7000 system supporting DOF 7-PO and it is different from the local system, a second DOF 7-PO connection called a connection-relay is initiated on the target system through the medium of the Network Exchange Handler. The Remote Administrative Exchange Handler (RAEH server) must be started when initiating a DOF 7-PO connection on a remote system.

6.17 Send Command (H_PMSSENDCMD)

Function

To send a command to a system or network object manager.

Format

```
#include <pmos.h>

{ h_pmssendcmd (pmsid,cmdid,param,station,every,site) }
 unsigned long pmsid;
 unsigned short cmdid;
 struct _pms_param param;
 char *station;          /* or char station[9]; */
 int every;
 char *site;             /* or char site[5];    */
```

Description of the structure

`pmsid`	This parameter identifies the DOF 7-PO connection.
`cmdid`	This parameter identifies the command and handles its related responses.
`param.descptr`	This parameter points to the descriptor of the command.
`param.descln`	This parameter is the length of the command descriptor.
`param.recptr`	This parameter points to the record containing the parameters of the command. When the related command has no parameter, `NULL_PTR` must be specified.
`param.recln`	This parameter is the length of the command record. When the command record does not exist, a null value must be given.

`station`	This parameter is the string containing the logical station specified in the command, and it is ignored if the format of the record is not the GCOS format. This station modifies the semantics of the command. The string must be null terminated. When the station is the local one, `PMS_LOCAL_STATION`, which is the constant defined in the <pmos.h> header file, must be specified.
`every`	This parameter submits the command at regular time intervals. This interval is given in seconds by the value of the every parameter. A value of -1 means by convention that no repetition is required. A response to a repeatable command that is not being delivered is refreshed by the following response as soon as it is available. In this way, a DOF 7-PO requestor is certain to receive the latest response.
`site`	This parameter is used to submit a command on a target system different from the default system specified at the connection initiation. This string must be null terminated and less than 5 characters long. To submit a command on the default target system, site must be set equal to `PMS_DEFAULT_SITE`, which is a constant string defined in the `<pmos.h>` header file.

Return codes

Normal	
`DONE`	The function completed normally.
Abnormal	
`ARGERR`	There is an error in a parameter.
`ARVIOL`	The command is not accessible in this environment.
`CDERR`	The structure of the command is incompatible with the descriptor that describes it.
`CDUNKN`	The command is unknown.
`DESCERR`	Incoherence related to the descriptor of the response.
`NMTCHERR`	The command record is not validated as needed by the syntax described in the command descriptor.

NOTOPEN	This DOF 7-PO connection is not initiated.
PROJUNKN	The project is unknown because it is not found in catalog.
PSWVIOL	There is a password violation.
RLBPUNKN	The relation billing/project is unknown because it is not found in catalog.
RSUNKN	The remote system is unknown.
SEQERR	The caller does not have the right to manage this connection.
SHUTDOWN	GCOS 7 is in the shutdown phase so the issuing of new commands is denied.
STTNUNKN	The station is unknown because it is not found in catalog.
SYSOV	The command cannot be taken into account for one of two reasons. Either there is an overflow on the number of outstanding commands which is restricted to 32, there is a space overflow in the file used by DOF 7-PO to store the chains of commands and their related responses.
USERUNKN	The user is unknown because it is not found in catalog.
WAOV	There is an overflow on the work area used to communicate with the DOF 7-PO server. The descriptor and the record of the command are too large.
WITHSYN	The request identifier has a synonym. The request identifier is the `pmsid/cmdid` pair has a synonym.

Comments

This function submits a command to a system belonging to the primary network. This command must belong to the current environment of the DOF 7-PO requestor. The access rights are derived from those of the user defined at the initiation of the connection. This allows the possibility of submitting a command for another user than the owner of the current step.

The submitter must associate an identifier to the command. This command identifier must be different from all the other identifiers relating to outstanding commands belonging to the same connection. The command identifier allows the submitter to handle the command and its related responses. The station parameter is used to define the name of the ROF station specified in the command, if any. When meaningful, it modifies the semantics of the command.

When the `every` parameter is not equal to -1, it repeats the command at a frequency that the value of the parameter gives in seconds. As soon as it is available, a subsequent response refreshes a response to a repeatable command that is not being delivered. In this way, the requestor is certain to receive the latest available response.

The repetition of a command stops at the termination of the connection, in the case of an explicit request by `h_pmsclose`, or in the case of normal or abnormal task termination by an implicit request.

Finally, the number of outstanding commands is restricted to 32. In the same way, the space of the various chains of command/responses in the DOF 7-PO queue file is allocated according to a quota so that the system tables cannot overflow through overuse by a single user. In the case of overflow, any submission of a command is refused until enough space is available.

When site specifies `PMS_DEFAULT_SITE`, a command is submitted on the system defined at the initiation of the DOF 7-PO connection. If the target system is different from the local system, a DOF 7-PO connection is created on the remote system if necessary (i.e. if the remote system is a DPS 7000 system supporting DOF 7-PO and if no command had yet been sent to this remote system by this DOF 7-PO connection). Then the command is sent through the medium of the Network Exchange Handler. The Remote Administrative Exchange Handler (RAEH server) must be started when submitting a command to a remote system.

A. Cross-Reference Tables

In this appendix, we have divided the commands and their related responses into three general 'areas of application': system, TDS, and network. This division has been made simply for convenience, and does not reflect any actual division that your application must follow.

This appendix contains the following set of tables:

A.1 Table of Available Commands	Use this table to look up the function of any command (either OMH format or DSAC format).
A.2 Table of System Commands and Responses	Use this table to look up the set of messages that can be received by each system command (OMH format).
A.3 Table of TDS Commands and Responses	Use this table to look up the set of messages that can be received by each TDS-related command (OMH format).
A.4 Table of Network Commands and Responses	This table lists the messages that can be received by the network commands (DSAC format).
A.5 Table of Network Unsolicited Messages	This table lists the network unsolicited messages that can be received (DSAC format).
A.6 Table of Messages Showing Layout	This table consists of a listing of the contents (text and variables) of the messages in the *Structured Records (OMH Format)* documents.

A.1 Table of Available Commands

Table A-1. The Commands Available to a DOF 7-PO Application (1/9)

Command	Meaning	Area of Application
ALNTC	Allow new TDS correspondent	TDS
ASEX	Associate executive	Network
CLTL	Close inline test	Network
CDP	Cancel dump	System
CJ	Cancel job	System
CKTXCONV	Check TX conversation	TDS
CLCPOOL	Close correspondent pool	TDS
CLTF	Close TDS file	TDS
CNDIM	Connect dimension	System
CNFUNC	Connect function	System
CNLD	Connect load	System
CO	Cancel output	System
COW	Cancel ouput	System
CRCL	Create terminal cluster	Network
CRDC	Create connection descriptor	Network
CRDIM	Create dimension	System
CRDV	Create terminal device	Network
CRFL	Create filter	Network
CRFLT	Create filter	System
CRFLTST	Create filter set	System
CRSB	Create statistic block	Network
CRSP	Create SAP address	Network
CTC	Cancel TDS correspondent	TDS
CTSPW	Cancel TDS spawn	TDS
DAAC	Display administrative correspondent	Network
DAAF	Display administrative function	Network
DAAG	Display administrative group	Network
DAAL	Display attributes addressing list	Network
DACB	Display cable	Network
DACC	Display channel connection	Network
DACD	Display attributes connection descriptor	Network
DACH	Display channel	Network
DACL	Display attributes terminal cluster	Network
DACO	Display attributes correspondent	Network
DACT	Display controller	Network

Table A-1. The Commands Available to a DOF 7-PO Application (2/9)

Command	Meaning	Area of Application
DADV	Display attributes terminal device	Network
DAFL	Display filter	Network
DAID	Display ISO session descriptor	Network
DAIS	Display ISO session	Network
DALC	Display logical connection	Network
DALG	Display log	Network
DAMB	Display mailbox	Network
DANC	Display attributes network connection	Network
DANR	Display attributes network route	Network
DANU	Display attributes network users	Network
DAOP	Display operator	Network
DAPC	Display attributes physical connection	Network
DAPL	Display physical line	Network
DAQD	Display queue	Network
DASA	Display subdomain	Network
DASB	Display attributes statistic block	Network
DASC	Display session control	Network
DASGMIR	Deassign MIRLOG file	System
DASGF	Display assigned files	System
DASP	Display attributes of SAP address	Network
DASR	Display session route	Network
DASS	Display session	Network
DASV	Display server	Network
DASY	Display system	Network
DATC	Display transport connection	Network
DATL	Display inline test	Network
DATS	Display transport station	Network
DAVC	Display attributes virtual circuit	Network
DAWM	Display attributes of welcome message	Network
DBSVR	Debug server	System
DC_OC	Display configuration outclass	System
DC_OUTDV	Display configuration outtdv	System
DCX	Display complex	System
DDIM	Display dimension	System
DDP	Display dump	System
DDTIME	Display deviation time	System

Table A-1. The Commands Available to a DOF 7-PO Application (3/9)

Command	Meaning	Area of Application
DFUNC	Display function	System
DIOC	Display I/O cache	System
DISDIM	Disconnect dimension	System
DISFUNC	Disconnect function	System
DISLD	Disconnect load	System
DJ	Display job/submitter	System
DJAS	Display JAS	System
DLD_ALL	Display load option=allout	System
DLD_CLDTLD	Display load option=cldtld	System
DLD_LOAD	Display load	System
DLD_OUT	Display load option=output	System
DLDIM	Delete dimension	System
DLFL	Delete filter	Network
DLFLT	Delete filter	System
DMB	Display member	System
DMIR	Display mirror mode	System
DMOT	Display MOT	System
DO	Display output	System
DOP	Display output parameters	System
DOW	Display output	System
DPROW	Dprint	System
DPT	Dump TDS	TDS
DR	Display request	System
DRMS	Display RMS	System
DSEX	Dissociate executive	Network
DSRV	Display service	System
DTDS	Display TDS	TDS
DTX	Display TX	TDS
DXLC	Display XLC	System
EJR	Enter job request	System
EXECT	Execute TDS	TDS
EXTL	Execute inline test	Network
FJ	Force job	System
FO	Force ouput	System
FSVRTO	Force server timeout	System
GAAC	Get attributes of administrative correspondent	Network
GAAF	Get attributes of administrative function	Network
GAAG	Get attributes of administrative group	Network
GAAL	Get attributes of addressing list	Network

Table A-1. The Commands Available to a DOF 7-PO Application (4/9)

Command	Meaning	Area of Application
GACB	Get attributes of cable	Network
GACC	Get attributes of channel connection	Network
GACH	Get attributes of channel	Network
GACO	Get attributes of correspondent	Network
GACT	Get attributes of controller	Network
GADV	Get attributes terminal device	Network
GAFL	Get attributes of filter	Network
GAID	Get attributes of ISO session descriptor	Network
GAIS	Get attributes of ISO session	Network
GALC	Get attributes of logical connection	Network
GALG	Get attributes of log	Network
GAMB	Get attributes of mailbox	Network
GANC	Get attributes network connection	Network
GANR	Get attributes of network route	Network
GANU	Get attributes of network users	Network
GAOP	Get attributes of operator	Network
GAPC	Get attributes of physical connection	Network
GAPL	Get attributes of physical line	Network
GAQD	Get attributes of queue	Network
GASA	Get attributes of subdomain	Network
GASB	Get attributes of statistic block	Network
GASC	Get attributes of session control	Network
GASP	Get attributes of SAP address	Network
GASR	Get attributes of session route	Network
GASS	Get attributes of session	Network
GASV	Get attributes of server	Network
GASY	Get attributes of system	Network
GATC	Get attributes of transport connection	Network
GATL	Get attributes of inline test	Network

Table A-1. The Commands Available to a DOF 7-PO Application (5/9)

Command	Meaning	Area of Application
GATS	Get attributes of transport station	Network
GAVC	Get attributes of virtual circuit	Network
GAWM	Get attributes of welcome message	Network
GHCC	Get history of channel connection	Network
GHEX	Get history executive	Network
GHIS	Get history of ISO session	Network
GHLC	Get history of logical connection	Network
GHNC	Get history of network connection	Network
GHPC	Get history of physical connection	Network
GHPL	Get history of physical line	Network
GHSS	Get history of session	Network
GHTC	Get history of transport connection	Network
GHTL	Get history of diagnostic control	Network
GHVC	Get history of virtual circuit	Network
HJ	Hold job	System
HO	Hold output	System
HOW	Hold output	System
LDTIQS	Load TDS IQS	TDS
LDTMEM	Load TDS memory	TDS
LSAC	List administrative correspondent	Network
LSAF	List administrative function	Network
LSAG	List administrative group	Network
LSAL	List addressing list	Network
LSCB	List cable	Network
LSCC	List channel connection	Network
LSCD	List attributes connection descriptor	Network
LSCH	List channel	Network
LSCL	List terminal cluster	Network
LSCO	List correspondent	Network
LSCT	List controller	Network
LSCPOOL	List correspondent pool	TDS

Table A-1. The Commands Available to a DOF 7-PO Application (6/9)

Command	Meaning	Area of Application
LSDP	List dump	System
LSDV	List terminal device	Network
LSEX	List executive	Network
LSFL	List filter	Network
LSFLT	List filter	System
LSID	List ISO session descriptor	Network
LSIS	List ISO session	Network
LSLC	List logical connection	Network
LSLG	List log	Network
LSMB	List mailbox	Network
LSNC	List network connection	Network
LSNR	List network route	Network
LSNU	List network users	Network
LSOP	List operator	Network
LSPC	List physical connection	Network
LSPL	List physical line	Network
LSQD	List queue	Network
LSSA	List subdomain	Network
LSSB	List statistic block	Network
LSSC	List session control	Network
LSSP	List SAP address	Network
LSSR	List session route	Network
LSSS	List session	Network
LSSY	List system	Network
LSTC	List transport connection	Network
LSTC	List TDS correspondent	TDS
LSTF	List TDS file	TDS
LSTS	List transport station	Network
LSTSPW	List TDS spawn	TDS
LSVC	List virtual circuits	Network
LSWS	List workstation	Network
MDC_OC	Modify configuration outclass	System
MDCPOOL	Modify correspondent pool	TDS
MDDIM	Modify dimension	System
MDFLT	Modify filter	System
MDIOC	Modify I/O cache	System
MDJ	Modify job	System
MDLD	Modify load	System
MDLD_SYS	Start/terminate load	System
MDMIR	Modify mirror mode	System
MDMOT	Modify MOT	System
MDOP	Modify output parameters	System
MDST_ROF	Modify station rescue_off	System
MDST_RON	Modify station rescue_on	System
MDTDS	Modify TDS	TDS

Table A-1. The Commands Available to a DOF 7-PO Application (7/9)

Command	Meaning	Area of Application
MDTIME	Modify time	System
MDTMOT	Modify TDS MOT	TDS
MDTRSO	Modify TDS restart options	TDS
MDTX	Modify TX	TDS
MDXLC	Modify XLC	System
MO	Modify output	System
NBAC	Number of administrative correspondents	Network
NBAF	Number of administrative functions	Network
NBAG	Number of administrative groups	Network
NBAL	Number addressing list	Network
NBCB	Number of cables	Network
NBCC	Number of channel connections	Network
NBCD	Number connection descriptor	Network
NBCH	Number of channels	Network
NBCL	Number terminal cluster	Network
NBCO	Number of correspondents	Network
NBCT	Number of controllers	Network
NBDV	Number terminal device	Network
NBFL	Number of filters	Network
NBIS	Number of ISO sessions	Network
NBLC	Number of logical connections	Network
NBLG	Number of logs	Network
NBMB	Number of mailboxes	Network
NBNC	Number of network connections	Network
NBNR	Number of network routes	Network
NBNU	Number network users	Network
NBPC	Number of physical connections	Network
NBOP	Number of operators	Network
NBPL	Number of physical lines	Network
NBQD	Number of queues	Network
NBSA	Number of subdomains	Network
NBSB	Number of statistic blocks	Network
NBSC	Number of session controls	Network
NBSP	Number SAP address	Network
NBSR	Number of session routes	Network
NBSS	Number of sessions	Network
NBSY	Number of systems	Network

Table A-1. The Commands Available to a DOF 7-PO Application (8/9)

Command	Meaning	Area of Application
NBTC	Number of transport connections	Network
NBTS	Number of transport stations	Network
NBVC	Number of virtual circuits	Network
NBWS	Number of workstations	Network
OCPOOL	Open correspondent pool	TDS
OTF	Open TDS file	TDS
PVNTC	Prevent new TDS correspondent	TDS
REPLY	Reply	System
RJ	Release job	System
RO	Release output	System
ROW	Release output	System
RSCMIR	Resync mirror shared	System
SCMSR	Start CMSR	System
SIOC	Start I/O cache	System
SIOS	Start I/O server	System
SLD	Start load	System
SMB	Start member	System
SNDT	Send TDS	TDS
SNDTU	Send TDS user	TDS
SOW	Start output writer	System
SPRVT	Supervise TDS	TDS
SRMS	Start RMS	System
SSRV	Start service	System
SWLG	Swap log	Network
TCMSR	Terminate CMSR	System
TIOC	Terminate I/O cache	System
TIOS	Terminate I/O server	System
TKMB	Takeover member	System
TLD	Terminate load	System
TMB	Terminate member	System
TOW	Terminate output writer	System
TRMS	Terminate RMS	System
TSRV	Terminate server	System
TSYS	Terminate system	System
TTDS	Terminate TDS	TDS
TTSV	Terminate server	Network
TXAC	Text broadcast to administrative correspondent	Network
TXAF	Text broadcast to administrative function	Network
TXDV	Text broadcast to terminal device	Network
TXEX	Text broadcast to executive	Network
TXLG	Text broadcast to log	Network

Table A-1. The Commands Available to a DOF 7-PO Application (9/9)

Command	Meaning	Area of Application
TXOP	Text broadcast to operator	Network
UNLDTIQS	Unload TDS IQS	TDS
UNLDTMEM	Unload TDS memory	TDS
UNLMIR	Unlock MIRLAB volume	System
UNLMIRF	Unlock MIRLOG file	System
UNPMIR	Unpair mirror volume	System
UPAC	Update administrative correspondent	Network
UPAF	Update administrative function	Network
UPAG	Update administrative group	Network
UPAL	Update addressing list	Network
UPCB	Update cable	Network
UPCD	Update connection descriptor	Network
UPCH	Update channel	Network
UPCL	Update terminal cluster	Network
UPCO	Update correspondent	Network
UPCT	Update controller	Network
UPDV	Update terminal device	Network
UPFL	Update filter	Network
UPLG	Update log	Network
UPNR	Update network route	Network
UPNU	Update network users	Network
UPOP	Update operator	Network
UPPL	Update physical line	Network
UPQD	Update queue	Network
UPSA	Update subdomain	Network
UPSB	Update statistic block	Network
UPSC	Update session control	Network
UPSP	Update SAP address	Network
UPSR	Update session route	Network
UPSV	Update server	Network
UPSY	Update system	Network
UPTL	Update inline test	Network
UPTS	Update transport station	Network
UPWM	Update welcome message	Network

A.2 Table of System Commands and Responses

Table A-2. System Commands and Responses (1/3)

System Command	Messages that may be received in response
CDP	DP04, DP10, DP11, DP12, DP13
CJ	OP66, PM51, SH04, SH09, SH11, SH13, SH26, SH45
CNDIM	AR45, AR47, AR70, AR71, AR73, AR90, AR91, AR92, AR95, AR96, AR99, PM51
CNFUNC	AR10, AR11, AR45
CNLD	OP66, PM51, SH03, SH10, SH31, SH43
CO	PM51, OU14, OU81, OU82, OU83, OU84, OU85, OU88, OU90
COW	OU01, OU05, OU80, OU81, OU82, OU83, OU84, OU85, OU88, OU90
CRDIM	AR35, AR45, AR70, AR79, AR80, AR81, PM51
CRFLT	FT01, FT02, FT05, PM51
CRFLTST	FT01, FT02, FT05, PM51
DASGMIR	MR12, MR13, MR19
DBSVR	TF05, TF06, TF07, TF50, TF51, TF52, TF53, TF54, TF55, TF56, TF57, TF58
DASGF	OP66, SH46, SH47, SH48, SH49
DC_OC	OU05, OU18, OU19, OU23, OU34, OU40
DC_OUTDV	OU05, OU06, OU11, OU19, OU38, OU40
DCX	CM03, CM19, CM21, CM26
DDIM	AR08, AR09, AR32, AR33, AR46, AR47, AR49, AR50, AR55, AR56, AR57, AR58, AR61, AR62, AR63, AR65, AR66, AR67, AR68, AR75, AR76, AR78, AR82, AR83, AR86, AR87, AR88, AR94, PM51
DDP	DP02, DP03, DP10, DP12
DDTIME	TM03, TM04, TM05, TM06, TM08, TM09, TM10, TM11, TM15, TM16, TM17, TM18
DFUNC	AR07, AR10, AR14, AR15
DIOC	CI02, CI03, CI05, CI50, CI51, CI52, CI53, CI54, CI99
DISDIM	AR42, AR43, AR44, AR45, AR47, AR70, AR73, AR90, AR97, AR98, AR99, PM51
DISFUNC	AR10, AR11, AR45
DISLD	OP66, PM51, SH03, SH10, SH31
DJ	OP66, PM51, SH09, SH14, SH15, SH17, SH19, SH20, SH28, SH42
DJAS	JP80
DLD_ALL	OU01, OU05, OU39
DLD_CLDTLD	SH32, SH40, SH41
DLD_LOAD	OP66, PM51, SH10, SH16, SH21, SH22, SH32, SH34, SH35
DLD_OUT	OU01, OU03, OU05, OU21

Table A-2. System Commands and Responses (2/3)

System Command	Messages that may be received in response
DLDIM	AR45, AR47, AR70, AR84, AR85, PM51
DLFLT	FT01, FT02, FT05, PM51
DMB	CM03, CM19, CM21, CM26
DMIR	MR13, MR16
DMOT	OP66, PM51, SH13, SH29
DO	PM51, OU01, OU03, OU08, OU09, OU12, OU20, OU21
DOP	OU44, OU82, OU88
DOW	OU01, OU05, OU08, OU09, OU12, OU20
DPROW	None
DR	OP66, PM03, PM04, PM51
DRMS	OP66, OP82, OP83, OP84, PM51
DSRV	CM03, CM19, CM23, CM26, CM36
DXLC	AR45, AR56, AR57, AR74, PM51
EJR	IN01, JB01, JB02, JB08, OP66, PM51
FO	OU82, OU84, OU85, OU86, OU88, OU90
FJ	OP66, PM51, SH09, SH11, SH13, SH23, SH26
FSVRTO	CM17, CM19, CM55
HJ	OP66, PM51, SH04, SH09, SH11, SH13, SH23, SH26, SH44, SH45
HO	PM51, OU81, OU82, OU83, OU84, OU85, OU88, OU90
HOW	OU01, OU05, OU80, OU81, OU83, OU84, OU85, OU88, OU90
LSDP	DP01, DP10, DP12
LSFLT	FT01, FT02, FT03, FT04, FT05, FT06, PM51
MDC_OC	OU01, OU05, OU80, OU81, OU82, OU83, OU84, OU85, OU88, OU90
MDDIM	AR45, AR47, AR48, AR70, AR73, AR74, AR93, PM51
MDFLT	FT01, FT02, FT05, PM51
MDIOC	CI02, CI03, CI05, CI06, CI08, CI09, CI10, CI11, CI13, CI14, CI99
MDJ	OP66, PM51, SH05, SH08, SH09, SH10, SH11, SH12, SH13, SH26, SH27, SH30
MDLD	OP66, PM51, SH03, SH05, SH08, SH10, SH13, SH34
MDLD_SYS	OP66, PM51, SH18
MDMIR	MR13, MR16
MDMOT	OP66, PM51, SH03, SH13
MDOP	None
MDST_ROF	OU05, OU06
MDST_RON	OU05, OU06
MDTIME	TM03, TM04, TM05, TM06, TM08, TM09, TM10, TM11, TM15, TM16, TM17, TM18
MDXLC	AR45, AR73, AR74, AR77, PM51

Table A-2. System Commands and Responses (3/3)

System Command	Messages that may be received in response
MO	OU01, OU05, OU80, OU81, OU82, OU83, OU84, OU85, OU88, OU90
REPLY	PM02, PM51
RJ	OP66, PM51, SH09, SH11, SH13, SH24, SH25, SH26
RO	PM51, OU81, OU82, OU83, OU84, OU85, OU88, OU90
ROW	OU01, OU05, OU80, OU81, OU83, OU84, OU85, OU88, OU90
RSCMIR	MR02, MR03, MR04, MR05, MR06, MR08, MR12, MR13, MR15
SCMSR	CM17, CM18, CM19
SIOC	CI01, CI03, CI04, CI05, CI06, CI08, CI10, CI13, CI99
SIOS	CI01, CI04, CI05, CI06, CI08, CI10, CI13, CI99
SLD	OP66, PM51, SH03, SH05, SH08, SH10, SH13, SH34
SMB	CM17, CM18, CM19, CM49
SOW	OUP1, OUP5, OUP6, OU63, OU66, OU67, OU68, OU69, OU70, OU71, OU72, OU73, OU77
SRMS	OP87, PM51
SSRV	CM17, CM18, CM19, CM20, CM65
TCMSR	CM17, CM18, CM19
TIOC	CI02, CI03 CI05, CI07, CI08, CI12, CI99
TIOS	CI02, CI03, CI05, CI06, CI07, CI08, CI12, CI99
TKMB	CM17, CM18, CM19, CM20
TLD	OP66, PM51, SH03, SH10, SH13, SH34
TMB	CM17, CM18, CM19, CM49
TOW	OUP5, OU63, OU64, OU68, OU69, OU70, OU71, OU74
TRMS	OP87, PM51
TSRV	CM17, CM18, CM19, CM20, CM65
TSYS	OP66, PM51, SH01, SH02, SH04
UNLMIR	MR12, MR13, MR18
UNLMIRF	MR12, MR13, MR17
UNPMIR	MR03, MR08, MR12, MR13, MR15, MR18

A.3 Table of TDS Commands and Responses

Table A-3. TDS Commands and Responses

TDS Command	Messages that may be received in response
ALNTC	TX51, TX54, TX55, TX56
CKTXCONV	TX54
CLCPOOL	TX56, TX71, TX72, TX73, TX74, TX75, TX80
CLTF	TX63, TX64
CTC	TX51, TX54, TX55, TX56
CTSPW	TX54, TX55, TX56
DPT	TX54, TX55
DTDS	TX04, TX05, TX06, TX07, TX09, TX10, TX20, TX21, TX30
DTX	TX08, TX11, TX12, TX13, TX14, TX15, TX16, TX19, TX25, TX52, TX57
EXECT	TX55, TX65, TX66
LDTIQS	TX54
LDTMEM	TX55, TX66, TX67
LSCPOOL	TX25, TX30, TX31, TX32, TX35, TX36, TX56, TX57
LSTC	TX22, TX23, TX24, TX25, TX26, TX27, TX28, TX29, TX33, TX34, TX56, TX57
LSTF	TX15, TX17, TX18, TX57
LSTSPW	TX22, TX25, TX38, TX57
MDCPOOL	TX56, TX71, TX72, TX73, TX74, TX75, TX80
MDTDS	TX54, TX55, TX61
MDTMOT	TX54
MDTRSO	TX54, TX55
MDTX	TX52, TX55, TX66
OCPOOL	TX56, TX70, TX71, TX72, TX73, TX74 TX75, TX80
OTF	TX63, TX64
PVNTC	TX54, TX55
SNDT	TX00, TX02, TX66
SNDTU	TX54, TX55
SPRVT	TX75, TX76, TX77
TTDS	TX53, TX54, TX55
UNLDTIQS	TX54
UNLDTMEM	TX55, TX67

A.4 Table of Network Commands and Responses

Network commands return one elementary response for each occurence of a network object. The responses have the same identifier as the commands. The DAMB (Display Mailboxes) command returns the response DAMB, the LSSY (List Systems) command returns the response LSSY, the GASS (Get Session Attributes) command returns the response GASS, and so on.

The following table shows the complete set of network commands and responses.

Table A-4. Network Commands and Responses

		Command/Responses Codes							
		AS	**CL**	**CR**	**DA**	**DL**	**DS**	**EX**	**GA**
	AC				xx				xx
	AF				xx				xx
	AG				xx				xx
	AL				xx				xx
	CB				xx				xx
	CC				xx				xx
	CD			xx	xx				xx
	CH				xx				xx
	CL			xx	xx				xx
	CO				xx				xx
	CT				xx				xx
	DV			xx	xx				xx
O	**EX**	xx					xx		
b	**FL**			xx	xx	xx			xx
j	**ID**				xx				xx
e	**IS**				xx				xx
c	**LC**				xx				xx
t	**LG**				xx				xx
	MB				xx				xx
C	**NC**				xx				xx
l	**NR**				xx				xx
a	**NU**				xx				xx
s	**OP**				xx				xx
s	**PC**				xx				xx
e	**PL**				xx				xx
s	**QD**				xx				xx
	SA				xx				xx
	SB			xx	xx				xx
	SC				xx				xx
	SP			xx	xx				xx
	SR				xx				xx

		AS	CL	CR	DA	DL	DS	EX	GA
	SS				xx				xx
	SV				xx				xx
	SY				xx				xx
	TC				xx				xx
	TL		xx		xx			xx	xx
	TS				xx				xx
	VC				xx				xx
	WM				xx				xx
	WS								

Command/Responses Codes

		GH	LS	NB	ST	SW	TT	TX	UP
	AC		xx	xx				xx	xx
	AF		xx	xx				xx	xx
	AG		xx	xx					xx
	AL		xx	xx					xx
	CB		xx	xx					
	CC	xx	xx	xx					
	CD		xx	xx					xx
	CH		xx	xx					xx
	CL		xx	xx					xx
	CO		xx	xx					xx
	CT		xx	xx					xx
	DV		xx	xx				xx	xx
O	**EX**	xx	xx					xx	
b	**FL**		xx	xx					xx
j	**ID**		xx						
e	**IS**	xx	xx	xx					
c	**LC**	xx	xx	xx					
t	**LG**		xx	xx		xx		xx	xx
	MB		xx	xx					
C	**NC**	xx	xx	xx					
l	**NR**		xx	xx					xx
a	**NU**		xx	xx					xx
s	**OP**		xx	xx				xx	xx
s	**PC**	xx	xx	xx					
e	**PL**	xx	xx	xx					xx
s	**QD**		xx	xx				xx	xx
	SA		xx	xx					xx
	SB		xx	xx					xx
	SC		xx	xx					xx
	SP		xx	xx					xx

		GH	LS	NB	ST	SW	TT	TX	UP
	SR		xx	xx					xx
	SS	xx	xx	xx					
	SV				xx		xx		xx
	SY		xx	xx					xx
	TC	xx	xx	xx					
	TL	xx							xx
	TS		xx	xx					xx
	VC	xx	xx	xx					
	WM								xx
	WS		xx	xx					

Please refer to the *DSAC User's Guide* for further information about object classes and command/response codes.

A.5 Table of Network Unsolicited Messages

The following table gives the unsolicited events (unsolicited messages) that may be received from network object managers.

Table A-5. Network Unsolicited Messages

		Unsolicited Event Codes												
		CL	**CS**	**ED**	**EM**	**ER**	**EX**	**FC**	**FD**	**FN**	**HR**	**OF**	**OP**	**XR**
	CC	xx				xx					xx	xx	xx	
	CH					xx								
	CL					xx								
	CT					xx								
	DV					xx								
	EX			xx	xx						xx			
	FT	xx				xx		xx	xx	xx	xx	xx	xx	
	FX	xx				xx						xx	xx	
	IS	xx									xx	xx	xx	
	LC	xx				xx					xx	xx	xx	
	LG	xx				xx						xx	xx	
	MB					xx								
	NA					xx								
	NC	xx				xx					xx	xx	xx	
	PC	xx				xx					xx	xx	xx	
	PL					xx					xx			
	SS	xx									xx	xx	xx	
	SU		xx											
	TC	xx				xx					xx	xx	xx	
	TL	xx				xx	xx				xx			xx
	TS					xx								
	VC	xx				xx					xx	xx	xx	
	WM					xx								

A.6 Table of Messages Showing Layout

This table lists all the messages that appear in *Structured Records (OMH Format) Part 2 - Messages.*

The elements of the messages are explained below:

`Lowercase`	variable
`UPPERCASE`	fixed text
`<<1>> or <<01>>`	variant number (many of the messages have several different variants)
`USM /*`	unsolicited message.

Note that the unsolicited messages included in this listing are those which have a structured record defined for them, and which appear in the *Structured Records (OMH Format)* manual. A DOF 7-PO application can however receive the same number of unsolicited messages as any other main operator.

Content of Messages

```
/*  AR07  << 1  >> dom  rjob func  */

/*  AR08  << 01  >> JOBTYPE  ASR(%)  CPU(%)  PWM(%)  */

/*  AR09  << 01  >> jobname asr . asrt cpu . cput max . maxt  */

/*  AR12  << 1  >> WARNING: SYSTEM ENTERS NORMAL USE  */
/*  AR12  << 2  >> SYSTEM BECOMES UNDERUSED  */

/*  AR32  << 1  >> CPU UTILIZATION IS TOO HIGH (%)     val  OVER  thv  */
/*  AR32  << 2  >> PAGING ACTIVITY IS TOO HIGH (%)     val  OVER  thv  */
/*  AR32  << 3  >> PAGE FAULT HANDLING IS TOO LONG (MS)     val  OVER  thv  */
/*  AR32  << 4  >> BKST IS ALMOST FULL (%)     val  OVER  thv  */
/*  AR32  << 5  >> BKST UTILIZATION IS TOO HIGH (%)     val  OVER  thv  */
/*  AR32  << 6  >> NUMBER OF SATURATED DISKS     val  OVER  thv  */
/*  AR32  << 7  >> NUMBER OF DISKS WITH QUEUES     val  OVER  thv  */
/*  AR32  << 8  >> PAGING UTILIZATION IS HIGH (%)     val  OVER  thv  */
/*  AR32  << 9  >> PAGE FAULT HANDLING IS LONG (MS)     val  OVER  thv  */
/*  AR32  << 10 >> BKST IS FULL (%)     val  OVER  thv  */

/*  AR33  << 01 >> DDIM MEANINGLESS: SPECIFIED RON SUSPENDED  */
```

```
/*  AR35  << 01 >> CRDIM  dimname  REJECTED: ORIGINAL PATTERN  likename  NOT
                   ALLOWED  */
USM /*  AR37  << 01 >> WARNING: ARM ENTERS DEGRADED MODE  */
USM /*  AR38  << 01 >> WARNING: SYSTEM ENTERS THRASHING  */
USM /*  AR39  << 01 >> SYSTEM LEAVES THRASHING  */

/*  AR42  << 01 >> DISDIM MEANINGLESS: ONLY JCG'S MAY BE DISCONNECTED FROM
                   dimname  */

/*  AR43  << 01 >> DISDIM MEANINGLESS: ONLY LM'S MAY BE DISCONNECTED FROM
                   dimname  */

/*  AR44  << 01 >> DISDIM REJECTED: USE LM/* IN COMMAND TO DISCONNECT A LM/* FROM
                   dimname  */

/*  AR45  << 01 >> UNKNOWN COMMAND  cdname  */

/*  AR46  << 01 >> DIMENSION:  dimname NO ACTIVE REGION.  */

/*  AR47  << 01 >> cdname  REJECTED: DIMENSION  dimname  UNKNOWN  */

/*  AR48  << 01 >> cdname COMPLETED. WARNING:CURRENT VALUE ENTERED FOR AT LEAST
                   ONE PARAMETER */

/*  AR49  << 01 >> ARM DEGRADED INTO BASIC.  rcode  */

/*  AR50  << 01 >> DIMNAME  RSR(%)   MPL  EX  ALLOC  WAIT  SUSP   SCH  REJ/MN  */


/*  AR55  << 01 >> DIMNAME  ASR(%)  ST/MN   ALLOC   IM/S  RT(S)      LF     XL  */

/*  AR56  << 01 >> XLC   ATTR   < HPRTYRANGE >  EPRTY   MAXCPU  */

/*  AR57  << 01 >> xlcval attr hprtymin hprtymax eprty cpumax  */

/*  AR58  << 01 >> DIMNAME  RELWGHT  <   MPL   >  <   XL   >  BWGHT  PA ICA HO  */


/*  AR61  << 01 >> DIMNAME  ASR(%)  CPU(%)  USED(MB)   FIX(MB)   IO/S  OUT/S
                   FLT/S  */

/*  AR62  << 1  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght Y   Y  Y  */

/*  AR62  << 2  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght Y   Y  N  */

/*  AR62  << 3  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght Y   N  Y  */

/*  AR62  << 4  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght Y   N  N  */

/*  AR62  << 5  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght N   Y  Y  */
/*  AR62  << 6  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght N   Y  N  */
```

```
/*  AR62  << 7  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght N   N  Y  */
/*  AR62  << 8  >> dimname relwght mnmpl mxmpl mnxl mxxl bwght N   N  N  */

/*  AR63  << 01 >> DIMENSION:  dimname CONNECTED OBJECTS: NONE  */

USM /*  AR64  << 01 >> ARM DEGRADED INTO BASIC.  rcode  */

/*  AR65  << 01 >> lm  */

/*  AR66  << 01 >> dimname rsr . rsrt mpl ex alloc wait susp sch rejf . rejft  */

/*  AR67  << 01 >> dimname asr . asrt stf . stft nbsteps imf . imft rt . rtt lf2 .
                   lf2t xl . xlt */

/*  AR68  << 01 >> dimname asr . asrt cpu . cput use . uset fix . fixt io . iot
                   out . outt flt. fltt  */

/*  AR69  << 1  >> dimnm rgid use . uset fix . fixt ru . rut lru . lrut flt . fltt
                   out . outt cur . curt nbj XXXXX  */


/*  AR70  << 01 >> cdname   dimname  COMPLETED  */

/*  AR71  << 01 >> CNDIM   dimname  COMPLETED. WARNING: AT LEAST A NAMED OBJECT
                      WAS ALREADY CONNECTED TO THIS DIM  */


/*  AR73  << 01 >> cdname  REJECTED: SYNTAX ERROR  */

/*  AR74  << 01 >> cdname  REJECTED: PARAMETER OUT OF RANGE  */

/*  AR75  << 1  >> DIMENSION:  dimname NO RONS.  */
/*  AR75  << 2  >> DIMENSION:  dimname ALL RONS SUSPENDED  */

/*  AR76  << 01 >> DIMNAME   RON   JC  ASR(%)  CPU(%)  USED(MB)  FIX(MB)
                   IO/S XL */

/*  AR77  << 01 >> MDXLC COMPLETED: XLC PARAMETER SET MODIFIED  */

/*  AR78  << 01 >> dimname X ron jc asr . asrt cpu . cput use . uset fix . fixt io
                   . iot xl . xlt  */

/*  AR79  << 01 >> CRDIM  dimname  REJECTED: ORIGINAL PATTERN likename UNKNOWN*/

/*  AR80  << 01 >> CRDIM  dimname  MEANINGLESS: NAMED DIMENSION ALREADY EXISTS  */

/*  AR81  << 01 >> CRDIM  dimname  UNSUCCESSFUL: TOO MANY DIMENSIONS  */

/*  AR82  << 01 >> DDIM REJECTED: UNKNOWN RUN OCCURENCE NUMBER  */
```

```
/*  AR83  << 01 >> ARM  bfull uselvl  USE.  */
/*  AR84  << 01 >> DLDIM dimname NOT ALLOWED: ATTEMPT TO DELETE A PERMANENT DIM */

/*  AR85  << 01 >> DLDIM  dimname  UNSUCCESSFUL: DIMENSION BUSY  */

/*  AR86  << 01 >> DIMNAME     MPL  ALLOC   USE(MB)   FIX(MB)  OUT/S   FLT/S  */

/*  AR87  << 01 >> dimname mpl nbsteps use . uset fix . fixt out . outt flt
                   . fltt  */

/*  AR88  << 01 >> DDIM  dimname  REJECTED: KEYWORD NOT ALLOWED IN BASIC  */



/*  AR90  << 01 >> cdname  dimname  REJECTED: AT LEAST A NAMED OBJECT CONNECTED TO
                   ANOTHER DIM */

/*  AR91  << 01 >> CNDIM  dimname  MEANINGLESS: ONLY JCG'S MAY BE CONNECTED  */

/*  AR92  << 01 >> CNDIM  dimname  MEANINGLESS: ONLY LM'S MAY BE CONNECTED  */

/*  AR93  << 01 >> MDDIM  dimname  REJECTED:  KEYWORD NOT ALLOWED IN BASIC  */

/*  AR93  << 02 >> MDDIM  dimname  REJECTED:  KEYWORD NOT ALLOWED FOR THIS
                   DIMENSION  */

/*  AR94  << 1  >> DIMNAME  CONNECTED OBJECTS  */

/*  AR94  << 2  >> NO OBJECT CONNECTED TO ANY DIMENSION  */

/*  AR95  << 01 >> CNDIM  dimname  UNSUCCESSFUL: LM CONNECTION TABLE WOULD
                   OVERFLOW  */

/*  AR96  << 01 >> CNDIM MEANINGLESS: ALL NAMED OBJECTS ALREADY CONNECTED TO DIM
                   dimname  */

/*  AR97  << 01 >> DISDIM dimname MEANINGLESS: ALL NAMED OBJECTS NOT CONNECTED  */

/*  AR98  << 01 >> DISDIM dimname COMPLETED. WARNING: AT LEAST A NAMED OBJECT WAS
                   NOT CONNECTED */

/*  AR99  << 01 >> cdname dimname COMPLETED. WARNING: AT LEAST AN OBJECT NAMED
                   TWICE IN COMMAND  */
```

```
/*  CI01  << 01 >> I/O CACHE ( sz  [MB]) STARTED AT  stime  */
/*  CI01  << 2  >> I/O  SERVER ( sz  [MB]) STARTED AT  stime  */

/*  CI02  << 1  >> I/O  CACHE  NOT STARTED  */
/*  CI02  << 2  >> I/O  SERVER  NOT STARTED  */
/*  CI03  << 1  >> I/O  CACHE  INITIALIZATION IN PROGRESS  */
/*  CI03  << 2  >> I/O  SERVER  INITIALIZATION IN PROGRESS  */

/*  CI04  << 1  >> I/O  CACHE  ALREADY STARTED  */
/*  CI04  << 2  >> I/O  SERVER  ALREADY STARTED  */

/*  CI05  << 1  >> I/O  CACHE  ACCESS RIGHTS VIOLATION  */
/*  CI05  << 2  >> I/O  SERVER  ACCESS RIGHTS VIOLATION  */

/*  CI06  << 1  >> I/O  CACHE  TERMINATION IN PROGRESS  */
/*  CI06  << 2  >> I/O  SERVER  TERMINATION IN PROGRESS  */

/*  CI07  << 1  >> I/O  CACHE  TERMINATED AT  stime  */
/*  CI07  << 2  >> I/O  SERVER  TERMINATED AT  stime  */

/*  CI08  << 01 >> FUNCTION ALREADY PERFORMED  */

/*  CI09  << 01 >> I/O CACHE SIZE MODIFIED AT: stime (sz [MB])  */

/*  CI10  << 01 >> NOT ENOUGH SPACE  */

/*  CI11  << 01 >> ATTEMPT TO REDUCE THE I/O CACHE SIZE UNDER 4 [MB]  */

/*  CI12  << 01 >> NOT SWAPPABLE WRITE-INTO IMAGES WERE LOST  */

/*  CI13  << 01 >> ILLEGAL VALUE FOR SIZE  */

/*  CI14  << 01 >> SIZE OUT OF RANGE  */

/*  CI50  << 1  >> BEGINNING DATE:  beg_date   beg_time CURRENT DATE  :  cur_date
                   cur_time INITIAL SIZE : beg_size  [MB] CURRENT SIZE  :
                   cur_size  [MB] STATUS :  ACTIVE  */

/*  CI50  << 2  >> BEGINNING DATE:  beg_date   beg_time CURRENT DATE  :  cur_date
                   cur_time INITIAL SIZE :  beg_size  [MB] CURRENT SIZE  :
                   cur_size  [MB] STATUS : SIZE BEING MODIFIED */

USM /*  CI50  << 3  >> BEGINNING DATE:  beg_date   beg_time CURRENT DATE  :
                       cur_date  cur_time INITIAL SIZE  :  beg_size  [MB] CURRENT
                       SIZE  : cur_size  [MB] STATUS : TERMINATION IN PROGRESS  */

USM /* CI51  << 1  >>       MODE          :  CACHE  */
USM /* CI51  << 2  >>       MODE          :  SIMULATION  */
```

```
USM /*  CI52 << 01 >>       GLOBAL STATISTICS SINCE  sdate  stime
   :   NB_IOS : NB_READS : NB_WRITES:  NB_HITS : %HITS
 --:----------:----------:----------:----------:-------:
       nb_ios : nb_reads : nb_writes: nb_hits  : p_hits  */

USM /*  CI53  << 01 >> CURRENT ACTIVITY (LAST 5 PERIODS):  PERIOD DURATION:
                       period  [SEC]*/

USM /*  CI54  << 01 >>  : NB_IOS  : NB_READS  : NB_WRITES : NB_HITS   : %HITS
                       1: nb1_ios : nb1_reads : nb1_rhits : nb1_whits : p1_hits
                       2: nb2_ios : nb2_reads : nb2_rhits : nb2_whits : p2_hits
                       3: nb3_ios : nb3_reads : nb3_rhits : nb3_whits : p3_hits
                       4: nb4_ios : nb4_reads : nb4_rhits : nb4_whits : p4_hits
                       5: nb5_ios : nb5_reads : nb5_rhits : nb5_whits : p5_hits */



/*  CI99  << 01 >> INTERNAL OR UNEXPECTED ERROR. COMMAND :  cmdnm ,MSGNB : msgtyp
                   RC :  gr4 */



USM /*  CM02  << 1  >> CM02  THE  HOLD OF THE SVRP SEMAPHORE  IS ABNORMAL  */
USM /*  CM02  << 2  >> CM02  THE  HOLD OF THE TSK SEMAPHORE IS ABNORMAL  */
USM /*  CM02  << 3  >> CM02  THE  INITIATON OF THE SVRP TASK IS ABNORMAL  */
USM /*  CM02  << 4  >> CM02  THE  INITIATION OF THE KERNEL TASK IS ABNORMAL  */
USM /*  CM02  << 5  >> CM02  THE  RELEASE OF THE TSK SEMAPHORE IS ABNORMAL  */
USM /*  CM02  << 6  >> CM02  THE  RELEASE OF THE SVRP SEMAPHORE IS ABNORMAL  */

USM /*  CM03  << 1  >> CM03  FAILURE IN THE  cmde  COMMAND RECEPTION,  rc  */
USM /*  CM03  << 2  >> CM03  FAILURE IN THE  cmde  COMMAND RECEPTION,  rc ,
                       RETRY LATER  */

USM /*  CM04  << 1  >> CM04  FAILURE IN THE RESPONSE EMISSION, rc  */
USM /*  CM04  << 2  >> CM04  FAILURE IN THE RESPONSE EMISSION, rc, RETRY LATER  */

USM /*  CM06  << 01 >> CM06  SERVICE  srv_name  STARTED ON MEMBER  mb_name  */

USM /*  CM07  << 01 >> CM07 SMD  DOF 7-PO SESSION IMPOSSIBLE TO OPEN, INTERNAL
                       ERROR NUMBER = number */

USM /*  CM08  << 01 >> CM08  CMSR STARTED  */

USM /*  CM09  << 01 >> CM09  CMSR TERMINATED  */

USM /*  CM15  << 01 >> CM15  IMPOSSIBLE TO START CMSR:  rc  */

/*  CM17  << 1  >> CM17   cmde  ACCEPTED  */
/*  CM17  << 2  >> CM17   cmde   element  ACCEPTED  */
```

```
/*  CM18  << 1  >> CM18   cmde   element1  MEANINGLESS : ALREADY STARTED  */
/*  CM18  << 2  >> CM18   cmde   element1  MEANINGLESS : ALREADY TERMINATED  */
/*  CM18  << 3  >> CM18   cmde   element1  MEANINGLESS : ALREADY IN PROCESS  */
/*  CM18  << 4  >> CM18   cmde   element1  MEANINGLESS : NO SERVICE TO TAKEOVER */
/*  CM18  << 5  >> CM18   cmde   MEANINGLESS : CMSC IN TERMINATION STEP   */
/*  CM18  << 6  >> CM18   cmde   MEANINGLESS : CMSC NOT YET OPERATIONAL   */
/*  CM18  << 7  >> CM18   cmde   MEANINGLESS : CMSC ALREADY IN TERMINATION STEP */
/*  CM18  << 8  >> CM18   cmde   element1   option1  MEANINGLESS :  cmde   option2
                   ALREADY IN PROCESS  */

/*  CM19  << 1  >> CM19   cmde  element  REJECTED :  element UNKNOWN  */
/*  CM19  << 2  >> CM19   cmde  element  REJECTED :  element STILL USED  */
/*  CM19  << 3  >> CM19   cmde  param  REJECTED : CMSR NOT RUNNING  */
/*  CM19  << 4  >> CM19   cmde  element1  REJECTED :  element2 NOT USED  */
/*  CM19  << 5  >> CM19   cmde  REJECTED : MEMBER NOT TERMINATED  */
/*  CM19  << 6  >> CM19   cmde  param  REJECTED : RESYNCHRONIZATION IN PROGRESS */
/*  CM19  << 7  >> CM19   cmde   element  REJECTED : RESYNC PARAM SPECIFIED FOR A
                   NON HA SERVICE  */
/*  CM19  << 8  >> CM19   cmde   element1  REJECTED :  element1  MUST BE STARTED
                   IN BACKUP MODE ON MEMBER  element2  WHEN RESYNC PARAM IS
                   SPECIFIED  */
/*  CM19  << 9  >> CM19   cmde   element1  REJECTED : TKMB element2  ALREADY IN
                   TERMINATION STAGE  */
/*  CM19  << 10 >> CM19   cmde  REJECTED : CMSC NOT YET OPERATIONAL  */
/*  CM19  << 11 >> CM19   cmde  REJECTED : CMSC IN TERMINATION STEP  */
/*  CM19  << 12 >> CM19   cmde   element1  REJECTED : SWITCHABILITY MODIFICATION
                   IMPOSSIBLE, MEMBER  element2  TELECOM ISOLATED  */
/*  CM19  << 13 >> CM19   cmde   element  REJECTED : SWITCHABILITY MODIFICATION
                   IMPOSSIBLE, AT LEAST ONE NON-HA STARTED SERVICE WHICH USES A
                   JAS SERVICE   */

/*  CM20  << 1  >> CM20   cmde   element1  REJECTED : USED_SERVICE element2
                   ACTIVE ON MEMBER element3  */
/*  CM20  << 2  >> CM20   cmde   element1  REJECTED : ACTIVE MEMBER  element2
                   TELECOM ISOLATION   */
/*  CM20  << 3  >> CM20   cmde   element1  REJECTED : ACTIVE MEMBER  element2
                   SILENT   */
/*  CM20  << 4  >> CM20   cmde   element1  REJECTED : MEMBER element2  TELECOM
                   ISOLATION   */
/*  CM20  << 5  >> CM20   cmde   element1  REJECTED : MEMBER element2  SILENT   */
/*  CM20  << 6  >> CM20   cmde   element1  REJECTED : MEMBER element2  NOT RUNNING
                   */
/*  CM20  << 7  >> CM20   cmde   element1  REJECTED : MEMBER element2
                   UNSWITCHABLE   */
/*  CM20  << 8  >> CM20   cmde   element1  REJECTED : MEMBER element2  STILL
                   RUNNING   */
/*  CM20  << 9  >> CM20   cmde   element1  REJECTED : TKMB element2  IN PROCESS */
/*  CM20  << 10 >> CM20   cmde   element1  REJECTED : MEMBER element2  NOT CRASHED
                   */
/*  CM20  << 11 >> CM20   cmde   element1  REJECTED : ACTIVE MEMBER  element2 NOT
                   RUNNING */
```

```
/*  CM20  << 12 >> CM20   cmde   element1  REJECTED : NOT MAPPED ON MEMBER
                   element2  */
/*  CM20  << 13 >> CM20   cmde   element1  REJECTED : ENTER TKMB WITH FORCE OPTION
                   TO GO ON */
/*  CM20  << 14 >> CM20   cmde   element1  REJECTED : ACTIVE MEMBER  element2  NOT
                   STARTED */
/*  CM20  << 15 >> CM20   cmde   element1  REJECTED : MEMBER element2  NOT STARTED
                   */
/*  CM20  << 16 >> CM20   cmde   element1  REJECTED : MEMBER element2  SWITCHABLE
                   */
/*  CM20  << 17 >> CM20   cmde   element1  REJECTED : RESYNC PARAMETER SPECIFIED
                   BUT NO DOUBLE FAILURE  */
/*  CM20  << 18 >> CM20   cmde   element1  REJECTED : MEMBER LIST INCOMPLETE  */

USM /*  CM21  << 1  >> srv_name srv_type svr_mod1 svr_mod2 usrv_list  */
USM /*  CM21  << 2  >> SERVICE   TYPE      EFFECTIVE_STATE  */
USM /*  CM21  << 3  >> SERVERS STATES UNKNOWN BECAUSE  member_name TELECOM
                       ISOLATED  */
USM /*  CM21  << 4  >> srv_name srv_type svr_eff_st  */
USM /*  CM21  << 5  >> usrv_list  */
USM /*  CM21  << 6  >> NO RUNNING SERVER   */

USM /*  CM22  << 1  >> CM22  MEMBER  member_name  SWITCHABLE  */
USM /*  CM22  << 2  >> CM22  MEMBER  member_name  UNSWITCHABLE  */
USM /*  CM22  << 3  >> CM22  MEMBER  member_name  TERMINATED  */
USM /*  CM22  << 4  >> CM22  MEMBER  member_name  TAKEN OVER  */
USM /*  CM22  << 5  >> CM22  MEMBER  member_name  ABNORMALLY TERMINATED  */
USM /*  CM22  << 6  >> CM22  MEMBER  member_name  CRASHED  */
USM /*  CM22  << 7  >> CM22  MEMBER  member_name  SILENT  */
USM /*  CM22  << 8  >> CM22  MEMBER  member_name  TELECOM ISOLATION  */
USM /*  CM22  << 9  >> CM22  MEMBER  member_name  NOT SILENT  */
USM /*  CM22  << 10 >> CM22  MEMBER  member_name  TELECOM CONNECTED  */
USM /*  CM22  << 11 >> CM22  MEMBER  member_name  STARTED  */

USM /*  CM23  << 1  >> mb_name :  svr_req_st1 svr_eff_st10  */
USM /*  CM23  << 2  >> mb_name :  svr_req_st1 svr_eff_st11  */
USM /*  CM23  << 3  >> mb_name :  svr_req_st1 svr_eff_st12  */
USM /*  CM23  << 4  >> mb_name :  svr_req_st1 svr_eff_st13  */

USM /*  CM24  << 1  >> CM24   FAILURE IN A  file_name  FILE WRITING ACCESS,
                       return_code , INTERNAL ERROR NUMBER =  number */
USM /*  CM24  << 2  >> CM24   FAILURE IN A  file_name  FILE READING ACCESS,
                       return_code , INTERNAL ERROR NUMBER =  number */
USM /*  CM24  << 3  >> CM24   FAILURE IN THE FILE  file_name OPENING,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM24  << 4  >> CM24   FAILURE IN THE FILE  file_name ASSIGNMENT,
                       return_code , INTERNAL ERROR NUMBER =  number  */
```

```
USM /*  CM25  << 1  >> CM25   FAILURE IN THE SMD SESSION OPENING,   return_code ,
                       INTERNAL ERROR NUMBER =  number  */
USM /*  CM25  << 2  >> CM25   FAILURE IN THE SMD SESSION CLOSING,  return_code ,
                       INTERNAL ERROR NUMBER = number  */
USM /*  CM25  << 3  >> CM25   FAILURE IN A SMD NOTIFICATION RECEPTION,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM25  << 4  >> CM25   FAILURE IN AN OPERATOR MESSAGE EMISSION,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM25  << 5  >> CM25   FAILURE IN AN OPERATOR RESPONSE EMISSION,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM25  << 6  >> CM25   FAILURE IN THE OPERATOR COMMAND RECEPTION,
                       return_code , INTERNAL ERROR NUMBER =  number  */

USM /*  CM26  << 1  >> CM26  COMPLEX  cx_name  :  SERVICE TYPE    mb_name1
                       mb_name2  USED_SERVICE  */
USM /*  CM26  << 2  >> REQUIRED_STATE   EFFECTIVE_STATE MONITORING_STATE
                       RUNNING_STATE  */
USM /*  CM26  << 3  >> REQUIRED_STATE        EFFECTIVE_STATE  */
USM /*  CM26  << 4  >> CM26  MEMBER  mb_name  :   */
USM /*  CM26  << 5  >> CM26  SERVICE  srv_name TYPE  srv_typ  : */
USM /*  CM26  << 6  >> mb_req_st mb_eff_st mb_mon_st mb_run_st  */
USM /*  CM26  << 7  >>     */

USM /*  CM27  << 1  >> CM27   FAILURE IN THE CREATION OF THE SEGMENT USED BY RWI,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM27  << 2  >> CM27   FAILURE IN THE CRASH IO INITIALIZATION,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM27  << 3  >> CM27   FAILURE IN THE RWI SESSION OPENING FOR THE
                       file_name  FILE, return_code , INTERNAL ERROR NUMBER =
                       number  */
USM /*  CM27  << 4  >> CM27   FAILURE IN THE EVA TABLE INITIALIZATION FOR THE
                       SERVICE TYPE srv_type  ,  return_code , INTERNAL ERROR
                       NUMBER =  number  */

USM /*  CM28  << 1  >> CM28   FAILURE IN NEW_ACTOR:
                       ACTOR TO  CREATE TYPE   =  actor1_type
                       CREATOR ACTOR    TYPE   =  actor2_type
                       CREATOR ACTOR  IDENTITY =   actor1_idnty
                       RETURN CODE             =  kernel_rc  */
USM /*  CM28  << 2  >> CM28   FAILURE IN SEND:     DESTINATION
                       ACTOR TYPE     =  actor1_type
                       DESTINATION ACTOR IDENTITY =  actor1_idnty
                       MESSAGE SENT TYPE          =  message_type
                       ISSUING ACTOR TYPE         =  actor2_type
                       ISSUING ACTOR IDENTITY     =  actor2_idnty
                       RETURN CODE                =  kernel_rc  */

USM /*  CM29  << 1  >> CM29    FILES KEYS  INCOHERENCE, INTERNAL ERROR NUMBER =
                       number  */
USM /*  CM29  << 2  >> CM29    CMSR INTERNAL TABLE  INCOHERENCE, INTERNAL ERROR
                       NUMBER =  number */
```

```
USM /*  CM30  << 01 >> CM30  COMPLEX GENERATION HAS NOT BEEN PERFORMED, INTERNAL
                       ERROR   NUMBER = number  */

USM /*  CM31  << 01 >> CM31  SERVER NOTIFICATION WITH AN INVALID ADMINISTRATION
                       KEY, USM INTERNAL ERROR NUMBER =  number  */

USM /*  CM32  << 1  >> CM32  SERVICE  service_name  TERMINATED ON MEMBER
                       member_name  */
USM /*  CM32  << 2  >> CM32  SERVICE  service_name  TERMINATED FORCE ON MEMBER
                       member_name  */
USM /*  CM32  << 3  >> CM32  SERVICE  service_name  ALREADY TERMINATED ON MEMBER
                       member_name  */

USM /*  CM33  << 1  >> CM33  SERVICE  service_name  ABORTED ON MEMBER  member_name
                       : RESTART ATTEMPT IN PROCESS  */
USM /*  CM33  << 2  >> CM33  SERVICE  service_name  ABORTED ON MEMBER  member_name
                       : IMPOSSIBLE TO RESTART  */

USM /*  CM34  << 1  >> CM34  SERVICE  service_name  RESTARTED ON MEMBER
                       member_name  */
USM /*  CM34  << 2  >> CM34  SERVICE  service_name  TAKEN OVER ON MEMBER
                       member_name  */

USM /*  CM35  << 1  >> CM35   TAKEOVER MEMBER  member_name  IN PROCESS  */
USM /*  CM35  << 2  >> CM35   RESYNCHRONISATION WITH THE CMSR OF THE MEMBER
                       member_name  HAS  SUCCEEDED : END OF THE DOUBLE FAILURE IF
                       THERE WAS ANY  */
USM /*  CM35  << 3  >> CM35   RESYNCHRONISATION WITH THE CMSR OF THE MEMBER
                       member_name  HAS  FAILED : IN THE CASE OF DOUBLE FAILURE,
                       PLEASE  REFER TO YOUR ADMINISTRATION GUIDE  */

USM /*  CM36  << 1  >> MAIN_SERVICE :  msrv_list1 , msrv_list2 , msrv_list3 ,
                       msrv_list4 , msrv_list5 , msrv_list6 , msrv_list7 ,
                       msrv_list8 , msrv_list9 , msrv_list10 , msrv_list11 ,
                       msrv_list12 , msrv_list13 , msrv_list14 , msrv_list15 ,
                       msrv_list16  */
USM /*  CM36  << 2  >> NO MAIN SERVICE  */
USM /*  CM36  << 3  >> USED_SERVICE :  usrv_list1 , usrv_list2 , usrv_list3 ,
                       usrv_list4 , usrv_list5 , usrv_list6 , usrv_list7 ,
                       usrv_list8 , usrv_list9 , usrv_list10 , usrv_list11 ,
                       usrv_list12 , usrv_list13 , usrv_list14 , usrv_list15 ,
                       usrv_list16  */
USM /*  CM36  << 4  >> NO USED SERVICE  */

USM /*  CM37  << 1  >> CM37   END OF RESYNCHRONIZATION  */
USM /*  CM37  << 2  >> CM37   END OF TAKEOVER PROCESSING  */
USM /*  CM37  << 3  >> CM37   WARNING : THE TIME IS NOT THE SAME ON THE TWO
                       MEMBERS, DATA INTEGRITY MAY BE DAMAGED IF A TAKEOVER OCCURS
                       */
```

```
USM /*  CM38  << 1  >> CM38   SERVICE  service_name  REFUSES TO SWITCH IN ACTIVE
                       MODE ON MEMBER member_name  */
USM /*  CM38  << 2  >> CM38   SWITCHABILITY MODIFICATION REQUESTED BY MEMBER
                       member_name  GIVEN UP : MEMBER  member_name  TELECOM
                       ISOLATED  */

USM /*  CM39  << 01 >> CM39  SERVICE  service_name  STARTED IN BACKUP MODE ON
                       MEMBER  member_name : USE THE TKMB COMMAND WITH THE FORCE
                       PARAMETER IF YOU WANT TO PUT IT IN THE ACTIVE MODE ON THIS
                       MEMBER  */

USM /*  CM40  << 01 >> CM40  ABNORMAL PROCESSING IN SERVICE service_name
                       TERMINATION ON MEMBER member_name ,   return_code : SERVER
                       CONSIDERED AS ABNORMALLY TERMINATED  */

USM /*  CM41  << 01 >> CM41  TIME OUT ON ACK_TERMINATED NOTIFICATION: SERVICE
                       service_name CONSIDERED AS ABNORMALLY TERMINATED ON
                       MEMBER  member_name  */

USM /*  CM42  << 01 >> CM42  CMSR IMPOSSIBLE TO RESTART, return_code  */

USM /*  CM43  << 01 >> CM43  NO SERVICE TO TAKEOVER  */

USM /*  CM44  << 01 >>  CM44  CRASH OF THE MEMBER  member_name  */

USM /*  CM45  << 1  >> CM45   cmde   param  REJECTED : RETRY LATER,  SWITCHABILITY
                       MODIFICATION IN PROGRESS  */
USM /*  CM45  << 2  >> CM45   cmde   param  REJECTED : RETRY LATER,  TAKEOVER IN
                       PROGRESS  */
USM /*  CM45  << 3  >> CM45   cmde   param  REJECTED : RETRY LATER,
                       TERMINATE_MEMBER IN PROGRESS  */
USM /*  CM45  << 4  >> CM45   cmde   param  REJECTED : RETRY LATER,  START_MEMBER
                       IN PROGRESS */
USM /*  CM45  << 5  >> CM45   cmde   param  REJECTED : RETRY LATER,  START_SERVICE
                       IN PROGRESS */
USM /*  CM45  << 6  >> CM45   cmde   param  REJECTED : RETRY LATER,
                       TERMINATE_SERVICE IN PROGRESS  */

USM /*  CM46  << 1  >> CM46 TKMB FORCE NO LONGER NECESSARY TO GO ON  */
USM /*  CM46  << 2  >> CM46 TKMB  option  INTERRUPTED : USE TKMB FORCE TO GO ON */
USM /*  CM46  << 3  >> CM46 TKMB  option  INTERRUPTED : MEMBER member_name TELECOM
                       ISOLATED, PLEASE REFER TO YOUR ADMINISTRATION GUIDE  */
USM /*  CM46  << 4  >> CM46 ENTER TKMB WITH FORCE OPTION IF THE MEMBER member_name
                       MUST BE CONSIDERED AS CRASHED  */

USM /*  CM47  << 01 >> CM47 SWITCHABILITY MODIFICATION INTERRUPTED: USE TKMB
                       FORCE TO GO ON  */

USM /*  CM48  << 01 >> CM48 DISK SURVEILLANCE INACTIVE WITH MEMBER  member_name */
```

```
/*  CM49  << 1  >> CM49   cmde   element  UNSUCCESSFULL: ISOLATED MEMBER  */
/*  CM49  << 2  >> CM49   cmde   element  UNSUCCESSFULL: CRASHED MEMBER  */

USM /*  CM50  << 1  >> CM50   FAILURE IN THE FILE   file_name CLOSING,
                       return_code, INTERNAL ERROR NUMBER =  number  */
USM /*  CM50  << 2  >> CM50   FAILURE IN THE FILE  file_name DEASSIGNMENT,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM50  << 3  >> CM50   FAILURE IN ONE OF THE FILE file_name  MULTI READING
                       ACCESSES, return_code , INTERNAL ERROR NUMBER = number  */

USM /*  CM51  << 01 >> CM51  FAILURE IN A MESSAGE EMISSION TO THE MEMBER
                       member_name , return_code , INTERNAL ERROR NUMBER =  number
                       */

USM /*  CM52  << 1  >> CM52   FAILURE IN THE DELETION OF THE SEGMENT USED BY RWI,
                       return_code , INTERNAL ERROR NUMBER =  number  */
USM /*  CM52  << 2  >> CM52   FAILURE IN THE RWI SESSION CLOSING FOR THE file_name
                       FILE, return_code , INTERNAL ERROR NUMBER =  number  */

USM /*  CM53  << 01 >> CM53  UNKNOWN OPERATOR COMMAND, INTERNAL ERROR NUMBER =
                       number  */

USM /*  CM54  << 01 >> CM54  SMD P DOF 7-PO SESSION CLOSED, return_code , INTERNAL
                       ERROR NUMBER = number  */

USM /*  CM55  << 1  >> CM55 TAKEOVER MEMBER member_name  GIVEN UP,  return_code */
USM /*  CM55  << 2  >> CM55 cmd   service_name  GIVEN UP: TAKEOVER IN PROGRESS  */
USM /*  CM55  << 3  >> CM55 cmd   service_name  UNSUCCESSFUL: NOT WAITING FOR A
                       SERVER NOTIFICATION  */

USM /*  CM56  << 1  >> CM56  FAILURE IN THE TAKEOVER RESTORING PROCESSING,
                       return_code , INTERNAL ERROR NUMBER =  number  */

USM /*  CM58  << 1  >> CM58  CMSR WILL WAIT UP TO 5 MINUTES FOR TELECOM TO BE
                       STARTED  */
USM /*  CM58  << 2  >> CM58  CMSR HAS STARTED WITHOUT TELECOM  */

USM /*  CM59  << 1  >> CM59   TAKEOVER MEMBER  member_name  IN PROCESS  */
USM /*  CM59  << 2  >> CM59   RESYNCHRONISATION WITH THE CMSR OF THE MEMBER
                       member_name  HAS SUCCEEDED : END OF THE DOUBLE FAILURE IF
                       THERE WAS ANY   */
USM /*  CM59  << 3  >> CM59   RESYNCHRONISATION WITH THE CMSR OF THE MEMBER
                       member_name  HAS FAILED: IN THE CASE OF A DOUBLE FAILURE,
                       PLEASE REFER TO YOUR ADMINISTRATION GUIDE  */

USM /*  CM61  << 1  >> CM61   cmde   element  UNSUCCESSFUL: SERVICE  element1
                       BACKUP SWITCHING REJECT ON MEMBER  element2  */
USM /*  CM61  << 2  >> CM61   cmde   element  UNSUCCESSFUL ON MEMBER  element2  */
USM /*  CM61  << 3  >> CM61   cmde   element1  UNSUCCESSFUL ON MEMBER  element2 :
                       element2  TELECOM ISOLATED  */
```

```
USM /*  CM61  << 4  >> CM61   cmde   element1  UNSUCCESSFUL ON MEMBER  element2 :
                       element2  NOT RUNNING */
USM /*  CM61  << 5  >> CM61   cmde   element  UNSUCCESSFUL ON MEMBER  element2  :
                       USED_SERVICE element1 STARTING REJECT ON MEMBER  element2
                       */
USM /*  CM61  << 6  >> CM61   cmde   element  UNSUCCESSFUL: SERVICES NOT ALL
                       TERMINATED  */
USM /*  CM61  << 7  >> CM61   cmde   element1  UNSUCCESSFUL: MEMBER  element2
                       UNSWITCHABLE  */
USM /*  CM61  << 8  >> CM61   cmde   element1  UNSUCCESSFUL ON MEMBER  element2 :
                       element2 NOT STARTED */
USM /*  CM61  << 9  >> CM61   cmde   element1  UNSUCCESSFUL : MEMBER  element2
                       TELECOM ISOLATION DURING THE SWITCHABILITY MODIFICATION,
                       RETRY LATER  */

USM /*  CM64  << 1  >> CM64   VCAM WORKSTATION FAILURE, INTERNAL REASON = sg1_rc,
                       INTERNAL ERROR   NUMBER = number  */
USM /*  CM64  << 2  >> CM64   VCAM MAILBOX FAILURE, INTERNAL REASON =  sg1_rc ,
                       INTERNAL ERROR   NUMBER  =  number  */
USM /*  CM64  << 3  >> CM64   FAILURE IN THE VCAM LINK WITH THE MEMBER
                       member_name , INTERNAL  REASON = sg1_rc , INTERNAL ERROR
                       NUMBER =  number  */

USM /*  CM65  << 01 >> CM65   cmde   element1  REJECTED : INVALID OPTION PARAMETER
                       VALUE  */

USM /*  CM66  << 1  >> CM66  SERVICE  service_name  ABORTED ON MEMBER  member_name
                       */
USM /*  CM66  << 2  >> CM66  SERVICE  service_type   service_name  STARTED IN
                       BACKUP MODE ON MEMBER member_name  */
USM /*  CM66  << 3  >> CM66  SERVICE  service_type   service_name STARTED IN
                       ACTIVE MODE ON MEMBER  member_name  */

USM /*  CM67  << 1  >> CM67  FAILURE OF THE  smd_order  SMD_ORDER FOR THE SERVICE
                       service_name , return_code, INTERNAL ERROR NUMBER =  number
                       */
USM /*  CM67  << 2  >> CM67  TIME OUT AFTER THE  smd_order SMD_ORDER EXECUTION FOR
                       THE SERVICE service_name  */

USM /*  CM68  << 1  >> CM68 CMSR RESYNCHRONIZATION IN PROGRESS  */
USM /*  CM68  << 2  >> CM68 END OF CMSR RESYNCHRONIZATION  */
USM /*  CM68  << 3  >> CM68 CMSR NOT TERMINATED : STILL RUNNING SERVERS  */

USM /*  CM69  << 1  >> CM69  FAILURE IN THE  HALOCK INITIALIZATION,  return_code ,
                       INTERNAL ERROR  NUMBER =  number  */
USM /*  CM69  << 2  >> CM69  FAILURE IN THE  HALOCK TERMINATION, return_code ,
                       INTERNAL ERROR  NUMBER = number  */
USM /*  CM69  << 3  >> CM69  FAILURE IN THE  HALOCK FUNCTION, return_code ,
                       INTERNAL ERROR NUMBER  = number  */
USM /*  CM69  << 4  >> CM69  FAILURE IN THE  HAUNLOCK FUNCTION, return_code ,
                       INTERNAL ERROR NUMBER  =  number  */
```

```
/*  DP01  << 01 >> dumpid PGID= pgid lmname date  */

/*  DP02  << 01 >> dumpid PGID= pgid lmname date TERMSG= termsg */

/*  DP03  << 01 >> percent % OF SYS.SPDUMP SPACE FREE    NBDUMP= nbdump  */

/*  DP04  << 01 >> dumpid PGID= pgid lmname CANCELED  */

USM /* DP05 << 01 >> dumpid PGID= pgid lmname DUMP CREATED  */

USM /* DP06 << 01 >> SYS.SPDUMP FULL, DUMP SWITCHED TO SYSOUT  */

USM /* DP07 << 01 >> UNABLE TO OPEN SYS.SPDUMP.  rcode SWITCH TO SYSOUT DUMP  */

USM /* DP08 << 01 >> UNABLE TO OPENS  dumpid  IN SYS.SPDUMP . rcode SWITCH TO
                     SYSOUT DUMP  */

USM /* DP09 << 01 >> UNABLE TO CLOSE  dumpid  IN SYS.SPDUMP . rcode  */

/*  DP10  << 01 >> UNABLE TO ACCESS SYS.SPDUMP.  rcode  */

/*  DP11  << 01 >> ILLEGAL ACCESS TO DUMP :  dumpid  */

/*  DP12  << 01 >> NO DUMP SELECTED  */

/*  DP13  << 01 >> SYNTAX ERROR  */

USM /* DP14 << 01 >> ERROR IN PUT ON SYS.SPDUMP .  rcode SWITCH TO SYSOUT DUMP  */

USM /* DP15 << 01 >> WARNING : 80% OF SYS.SPDUMP SPACE USED  */



/*  FT01  << 1  >> ALL OR SET MUST BE SPECIFIED  */
/*  FT01  << 2  >>  */
/*  FT01  << 3  >> INVALID KEY  */
/*  FT01  << 4  >> PRTY_MIN IS MANDATORY WITH PRTY_MAX  */
/*  FT01  << 5  >> PRTY_MAX MUST BE GREATER THAN PRTY_MIN  */
/*  FT01  << 6  >> ONE FILTERING CRITERION AT LEAST MUST BE SPECIFIED  */
/*  FT01  << 7  >> KEY_MAX MUST BE GREATER THAN KEY_MIN  */
/*  FT01  << 8  >> ALL AND FILTER ARE EXCLUSIVE  */
/*  FT01  << 9  >> RON MUST BE A DECIMAL NUMBER  */

/*  FT02  << 1  >> MAX NUMBER OF FILTER_SETS HAS BEEN REACHED  */
/*  FT02  << 2  >> FILTER TABLE OVERLOAD  */
/*  FT02  << 3  >> FILTER_SET ALREADY CREATED  */
/*  FT02  << 4  >> FILTER_SET CREATED  */
/*  FT02  << 5  >> INVALID MESSAGE KEY  msgkey  */
/*  FT02  << 6  >> FILTER CREATED  */
/*  FT02  << 7  >> UNKNOWN FILTER_SET NAME  */
/*  FT02  << 8  >> FILTER ALREADY CREATED  */
```

```
/*  FT02  << 9  >> FILTER_SET MODIFIED  */
/*  FT02  << 10 >> FILTER MODIFIED  */
/*  FT02  << 11 >> UNKNOWN FILTER NAME  */
/*  FT02  << 12 >> FILTER_SET DELETED  */
/*  FT02  << 13 >> FILTER DELETED  */
/*  FT02  << 14 >> NO FILTER_SET  */
/*  FT02  << 15 >> MAX NUMBER OF FILTERS HAS BEEN REACHED  */
/*  FT02  << 16 >> ALL FILTER_SET(S) DELETED  */

/*  FT03  << 01 >> fltst  fltstate msgprty  */

/*  FT04  << 1  >> flt  fltst  fltstate flttyp msgkey1  msgkey2 msgprty1  msgprty2
                   msgtyp  jnm ron  */
/*  FT04  << 2  >> flt  fltst  fltstate flttyp msgkey1  msgkey2 msgprty1
                   msgprty2  msgtyp  jnm*/

/*  FT05  << 1  >> gr4 COMMAND  CRFS NOT PERFORMED  */
/*  FT05  << 2  >> gr4 COMMAND  CRFLT NOT PERFORMED  */
/*  FT05  << 3  >> gr4 COMMAND  MDFLT NOT PERFORMED  */
/*  FT05  << 4  >> gr4 COMMAND  DLFLT NOT PERFORMED  */
/*  FT05  << 5  >> gr4 COMMAND  LSFLT NOT PERFORMED  */

/*  FT06  << 01 >> KEY PRIORITY TYPE JOBID RON  */



/*  IN01  << 1 >> ron IN jnm USER=usernm CLASS=jclass SPR=spr l_32  */
/*  IN01  << 2 >> ron HOLD jnm USER=usernm CLASS=jclass SPR=spr l_32  */



/*  JB01  << 01 >> ron STARTED jnm usernm jclass  */

/*  JB02  << 1 >> ron.ssn COMPLETED jnm usernm  */
/*  JB02  << 2 >> ron.ssn KILLED jnm usernm  */
/*  JB02  << 3 >> ron.ssn ABORTED jnm usernm  */
/*  JB02  << 17 >> ron.ssn COMPLETED jnm usernm jclass  */
/*  JB02  << 18 >> ron.ssn KILLED jnm usernm jclass  */
/*  JB02  << 19 >> ron.ssn ABORTED jnm usernm jclass SEV sevnb  <JB51>  */
/*  JB02  << 20 >> ron.ssn ABORTED jnm usernm jclass SEV sevnb=statustp  <JB51> */

USM /*  JB08 << 01 >> ron . ssn STEP  lm_nm  XPR= dpr PGID= jnb */



USM /*  MR01  << 1  >> dvnm PREMOUNTED  volnm  MIRROR PRIM  */
USM /*  MR01  << 2  >> dvnm PREMOUNTED  volnm  MIRROR NSTD  */
USM /*  MR01  << 3  >> dvnm PREMOUNTED  volnm  MIRROR ALONE  */
USM /*  MR01  << 4  >> dvnm PREMOUNTED  volnm  MIRROR PRIM WILL BE ALONE  */
USM /*  MR01  << 5  >> dvnm PREMOUNTED  volnm  MIRROR WILL BE COPIED  */
USM /*  MR01  << 6  >> dvnm PREMOUNTED  volnm  MIRROR WILL BE REFILLED  */
```

```
USM /*  MR01  << 7  >> dvnm PREMOUNTED  volnm  MIRROR SEC WILL BE NSTD  */
USM /*  MR01  << 8  >> dvnm PREMOUNTED  volnm  MIRROR PRIM RESYNCHRONIZED  */
USM /*  MR01  << 9  >> dvnm PREMOUNTED  volnm  MIRROR SEC RESYNCHRONIZED  */
USM /*  MR01  << 10 >> dvnm PREMOUNTED  volnm  MIRROR SEC  */
USM /*  MR02  << 1  >> RESYNCHRONIZATION OF  volnm   dvc  FAILED  */
USM /*  MR02  << 2  >> RESYNCHRONIZATION OF  volnm   dvc  SUCCESSFUL  */

USM /*  MR03  << 1  >> INVALIDATION OF  volnm   dvc  FAILED  */
USM /*  MR03  << 2  >> INVALIDATION OF  volnm   dvc  IN PROGRESS */
USM /*  MR03  << 3  >> INVALIDATION OF  volnm   dvc  REQUESTED BY SYSTEM  */
USM /*  MR03  << 4  >> INVALIDATION OF  volnm   dvc  ALREADY IN PROGRESS  */
USM /*  MR03  << 5  >> INVALIDATION OF  volnm   dvc  REQUESTED BY COMMAND  */
USM /*  MR03  << 6  >> INVALIDATION OF  volnm   dvc  REQUESTED BY REMOTE SYSTEM */

USM /*  MR04  << 1  >> RESYNCHRONIZATION OF SHARED VOLUMES FAILED */
USM /*  MR04  << 2  >> RESYNCHRONIZATION OF SHARED VOLUMES  COMPLETED  */
USM /*  MR04  << 3  >> RESYNCHRONIZATION OF SHARED VOLUMES  PARTLY COMPLETED  */
USM /*  MR04  << 4  >> RESYNCHRONIZATION OF SHARED VOLUMES  MEANINGLESS: ALREADY
                       STARTED  */

USM /*  MR05  << 1  >> volnm   dvc  NOT RESYNCHRONIZED: VOLUME NOT MIRROR  */
USM /*  MR05  << 2  >> volnm   dvc  NOT RESYNCHRONIZED: VOLUME NOT SHARED  */

USM /*  MR06  << 1  >> SITE.MIRLOG NOT ACCESSIBLE: DEGRADED MODE */
USM /*  MR06  << 2  >> SITE.MIRLOG NOT ACCESSIBLE: DEGRADED MODE WITHOUT POSSIBLE
                       ACCESS TO THE NSTDTAB  */

USM /*  MR07  << 1  >> MIR FUNCTIONALITY NOT PURCHASED  */
USM /*  MR07  << 2  >> WARNING: RESYNCHRONIZATIOON OF MIRROR DISKS IS NOT POSSIBLE
                       AT NEXT GCOS RESTART  */

USM /*  MR08  << 1  >> dvnm volnm  MIRROR ALONE  */
USM /*  MR08  << 2  >> dvnm volnm  INVALIDATED BY SYSTEM  */
USM /*  MR08  << 3  >> dvnm volnm  MIRROR PRIM NOT RESYNCHRONIZED: DEVICE STANDBY
                       */
USM /*  MR08  << 4  >> dvnm volnm  MIRROR SEC NOT RESYNCHRONIZED: DEVICE STANDBY
                       */

USM /*  MR09  << 1  >> dvnm ILLEGAL SYSTEM DISK,  MIRROR ATTRIBUTE IS FORBIDDEN */
USM /*  MR09  << 2  >> dvnm ILLEGAL BKST,  MIRROR ATTRIBUTE IS FORBIDDEN  */

USM /*  MR10  << 1  >> REFILLING  JOB CANNOT BE ENQUEUED FOR  volnm   dvc  */
USM /*  MR10  << 2  >> REFILLING  ACTION FAILED  */
USM /*  MR10  << 3  >> REFILLING  JOB ENQUEUED FOR  volnm  dvc  */

USM /*  MR11  << 1  >> dvnm DISMOUNT  volnm : MIRROR WITH OBSOLETE PAIRING DATE */
USM /*  MR11  << 2  >> dvnm DISMOUNT  volnm : A NON-MIRROR VOLUME HAS BEEN FOUND
                       */
USM /*  MR11  << 3  >> dvnm DISMOUNT  volnm :  INCONSISTENT MIRROR STATE  */
```

```
/*  MR12  << 1  >> cmdnm  ACCEPTED  */
/*  MR12  << 2  >> cmdnm  REJECTED  */
/*  MR12  << 3  >> cmdnm  SUCCESSFULL  */
/*  MR12  << 4  >> cmdnm  FAILED  */
/*  MR12  << 5  >> cmdnm  REJECTED : MIRROR DISKS NOT AUTHORIZED ON THIS SITE */
/*  MR13  << 1  >> FAILURE IN THE  cmdnm  COMMAND RECEPTION, gr4 , RETRY LATER  */
/*  MR13  << 2  >> cmdnm  FAILED,  gr4 , RETRY LATER  */
/*  MR13  << 3  >> FAILURE WHILE ISSUING REPLY TO THE cmdnm COMMAND,  gr4, RETRY
                   LATER  */

USM /*  MR15  << 1  >> INVALIDATION OF  THE SECONDARY COPY RELATED TO THE  volnm
                       dvc  ON THE OTHER SYSTEM REQUIRED  */

/*  MR16  << 1  >> MANUAL MIRROR MODE  */
/*  MR16  << 2  >> AUTOMATIC MIRROR MODE  */

USM /*  MR17  << 1  >> WARNING: RECAREA OF THE SITE.MIRLOG IS LOCKED BY THE OTHER
                       SYSTEM  THE UNLMIRF (GCL) OR UMIRF (OCL) COMMAND IS
                       REQUIRED  */
USM /*  MR17  << 2  >> WARNING: NSTDTAB OF THE SITE.MIRLOG IS LOCKED BY THE OTHER
                       SYSTEM  THE UNLMIRF (GCL) OR UMIRF (OCL) COMMAND IS
                       REQUIRED  */
USM /*  MR17  << 3  >> UNLOCK OF THE RECAREA PART OF THE SITE.MIRLOG SUCCESSFUL */
USM /*  MR17  << 4  >> UNLOCK OF THE NSTDTAB PART OF THE SITE.MIRLOG SUCCESSFUL */
USM /*  MR17  << 5  >> UNLOCK OF THE RECAREA PART OF THE SITE.MIRLOG FAILED  */
USM /*  MR17  << 6  >> UNLOCK OF THE NSTDTAB PART OF THE SITE.MIRLOG FAILED  */

USM /*  MR18  << 1  >> WARNING: MIRLAB OF  volnm   dvc  IS LOCKED BY THE OTHER
                       SYSTEM    THE  UNLMIR (GCL) OR UMIRV (JCL) COMMAND IS
                       REQUIRED  */
USM /*  MR18  << 2  >> UNLOCK OF MIRLAB OF  volnm   dvc  SUCCESSFUL  */
USM /*  MR18  << 3  >> UNLOCK OF MIRLAB OF  volnm   dvc  FAILED  */
USM /*  MR18  << 4  >> UNLOCK OF MIRLAB OF  volnm   dvc  MEANINGLESS  */

/*  MR19  << 1  >> INITIALIZATION OF THE SITE.MIRLOG SUCCESSFULL */
/*  MR19  << 2  >> INITIALIZATION OF THE SITE.MIRLOG FAILED  */
/*  MR19  << 3  >> DEASSIGNMENT OF THE SITE.MIRLOG SUCCESSFULL  */
/*  MR19  << 4  >> DEASSIGNMENT OF THE SITE.MIRLOG FAILED  */



USM /*  MU00  << 1  >> ron.ssn TDS :  tds  WARNING, REASON :  msgtxt gr4  */
USM /*  MU00  << 2  >> ron.ssn TDS :  tds  RESTARTABLE ABORT,
                       REASON :  msgtxt gr4  */
USM /*  MU00  << 3  >> ron.ssn TDS :  tds  FATAL ABORT, REASON :  msgtxt gr4  */
```

```
USM /*  MV00  << 1  >> ron.ssn TDS :  tds  WARNING, REASON :  msgtxt gr4  */
USM /*  MV00  << 2  >> ron.ssn TDS :  tds  RESTARTABLE ABORT,
                       REASON :  msgtxt gr4  */
USM /*  MV00  << 3  >> ron.ssn TDS :  tds  FATAL ABORT, REASON :  msgtxt gr4  */



/*  OP66  << 01 >>  */

/*  OP82  << 1  >> DRMS CONFIGURATION MODE ON SP :  */
/*  OP82  << 2  >> session  SESSION NOT INSTALLED  */
/*  OP82  << 3  >> session  SESSION INSTALLED  */
/*  OP82  << 4  >> session  SESSION INSTALLED MANUAL  */
/*  OP82  << 5  >> session  SESSION INSTALLED AUTOMATIC NO RECALL  */
/*  OP82  << 6  >> session  SESSION INSTALLED AUTOMATIC WITH RECALL  */
/*  OP82  << 7  >> OUTGOING CALL VALIDATION= validation  */
/*  OP82  << 8  >> RMS EXECUTION MODE ON SP :  mode  */

/*  OP84  << 01 >> RMS TRANSFER OUT STATUS : NUMBER OF BUFFERS BUSY =  nb_busy_buf
                   MAXIMUM NUMBER OF BUFFERS =  nb_max_buf  */

/*  OP87  << 1  >> SRMS ON= type  ENABLE= class  PERFORMED  */
/*  OP87  << 2  >> TRMS ON= type  LOCK= class  PERFORMED  */
/*  OP87  << 3  >> SRMS ON= type  ENABLE= class  REJECTED: ILLEGAL RMS STATE  */
/*  OP87  << 4  >> TRMS ON= type  LOCK= class  REJECTED: ILLEGAL RMS STATE  */
/*  OP87  << 5  >> DRMS  REJECTED: FUNCTION NOT AVAILABLE  */



/*  OU01  << 1 >> cmdnm ILLEGAL SYNTAX  */
/*  OU01  << 2 >> cmdnm DEVICE UNKNOWN  */
/*  OU01  << 3 >> cmdnm NON FORCED OUTPUT  */
/*  OU01  << 4 >> cmdnm OWQ OVERFLOW FOLLOWING INFORMATION IS FROM THE FIRST OWQ
                  TABLE. TYPE:     DO ALL FOR ALL OUTPUTS  */
/*  OU01  << 5 >> cmdnm QUEUE IS EMPTY  */
/*  OU01  << 6 >> cmdnm STATION=stnnm.substation */
/*  OU01  << 7 >> cmdnm NO OUTPUT  */
/*  OU01  << 8 >> cmdnm UNKNOWN JOB  */
/*  OU01  << 9 >> cmdnm OVERFLOW ON OUTPUT WRITERS TABLE  */

USM /*  OU02  << 01 >> dvnm OUTPUT ron : ouseqnb RESTART FROM?
                   (CURRENT PAGE NUMBER IS current_page) */

/*  OU03  << 01 >> DLD freesp_pct % OF SYSOUT SPACE FREE  */

/*  OU04  << 01 >>  GCOS: INVALID REPLY: ourstrep */

/*  OU05  << 01 >>  stnnm STATION UNKNOWN */
```

```
/*  OU06 << 1 >> cmdnm dvnm MEANINGLESS */

/*  OU06 << 2 >> cmdnm dvnm DEVICE NOT AVAILABLE */

/*  OU06 << 3 >> cmdnm dvnm STARTED */

/*  OU06 << 4 >> cmdnm dvnm MEANINGLESS: NO DEVICE FOUND */

/*  OU06 << 5 >> cmdnm dvnm MEANINGLESS: DEVICE NOT FOUND */

/*  OU06 << 6 >> cmdnm dvnm MEANINGLESS: DEVICE UNKNOWN */

USM /*  OU07  << 01 >>  dvnm TERMINATED BY SHUTDOWN */

/*  OU08  << 01 >> DO ron : ouseqnb ounm ouprty ouclass dvtyp oustate LINES=oulnnb
                   PAGES=oupgnb USER=usernm ORG_SYST=org_syst <OU62> <OU61> <OU60>
                   DEVCLASS=dvc MEDIA=md */

/*  OU09  << 1 >> cmdnm ron NO OUTPUT  */
/*  OU09  << 2 >> cmdnm ron UNKNOWN JOB  */
/*  OU09  << 3 >> cmdnm ron STATE OF JOB IS IN OR SCH  */

/*  OU11  << 01 >>  cmdnm NO CLASS STARTED ON dvnm */

/*  OU12  << 1 >> cmdnm UNKNOWN OUTPUT  */
/*  OU12  << 2 >> cmdnm OUTPUT BEING PROCESSED  */
/*  OU12  << 3 >> cmdnm OUTPUT ALREADY HELD  */
/*  OU12  << 4 >> cmdnm OUTPUT NOT IN QUEUE  */

USM /*  OU13  << 01 >> ron.ssn SYS.OUT OVERFLOW  */

USM /*  OU14  << 01 >> ron OUTPUT COMPLETED jnm usernm  ON DEVICE dvnm*/
    /*  OU14  << 02 >> ron OUTPUT COMPLETED jnm usernm  ON SITE sitenm*/

USM /*  OU16  << 01 >>  SYS.OUT RECOVER? */

/*  OU18  << 01 >> cmdnm DEFAULT PRTY = ouprty MEDIA = volnm LINES = ousz FOR CLASS ouclass

/*  OU19  << 01 >>  WRONG RETURN CODE IN OUTPUT WRITER PROCESSING gr4 */

/*  OU20  << 01 >> DO  <OU58>  <OU58>  LINES= oulnnb PAGES= oupgnb */

/*  OU21  << 01 >> DLD HOLD= hold  WAIT= wait  OUT= out  */

/*  OU23  << 01 >>  DC OC dvnm STARTED ON ou_class_lst DEVCLASS: dvc */

USM /*  OU25  << 1 >>  DYNAMIC EXTENSION OF SYSOUT PERFORMED occup_pct % OF SYSOUT SPACE
                       USED */

    /*  OU25  << 2 >>  DYNAMIC EXTENSION OF SYSOUT IMPOSSIBLE occup_pct % OF SYSOUT SPACE USED
                       gr4 */
```

```
USM /*  OU26  << 1  >>  dvnm l_16 CANCELLED gr4 EFN=efn subfile=sfn  */

USM /*  OU27  << 01 >>  dvnm l_16_01 l_16_02 l_32 gr4 sysout */

USM /*  OU28  << 01 >>  dvnm WRITER TERMINATED gr4 */

USM /*  OU29  << 01 >>  WARNING: IFN ifn IS NOT IN SYSOUT FILE FORMAT 1_8  EDITION PARAMETERS
                        WILL NOT BE TAKEN INTO ACCOUNT FOR FURTHER OUTPUT REQUESTS OF THIS
                        FILE */

USM /*  OU30  << 01 >> OVERFLOW ON OUTPUT WRITER CONTROL STRUCTURES  */

USM /*  OU31  << 01 >>  ERROR IN OUTPUT STATEMENT : l_8 gr4 */

USM /*  OU32  << 1 >>  OUTPUT WILL BE RESTARTED FROM BEGINNING */

    /*  OU32  << 2 >>  OUTPUT WILL BE RESTARTED FROM CURRENT ADDRESS */

    /*  OU32  << 3 >>  OUTPUT WILL BE RESTARTED FROM BACK= oupgnb */

USM /*  OU33  << 01 >>  dvnm OUTPUT ron : ouseqnb RESTART FROM? (CURRENT PAGE NUMBER IS
                        current_page ) */

/*  OU34  << 01 >>  UNABLE TO ACCESS STATION stnnm gr4 MAIN STATION ASSUMED

USM /*  OU35  << 01 >>  UNSUCCESSFUL SYS.OUT RECOVERY */

USM /*  OU36  << 01 >>  SSF ASSUMED */

USM /*  OU37  << 01 >>  dvnm output RENAMING NOT PERFORMED */

/*  OU38  << 1 >>  cmdnm dvnm STARTED ON CLASSES: ou_class_lst DEVCLASS: dvc STATION: stnnm ON
                   FILE: l_dollar sitenm l_2points_3 efn l_points sfn l_2points_1 md
                   l_2points_2 file_dvc */

/*  OU38  << 2 >>  cmdnm dvnm STARTED ON CLASSES: ou_class_lst DEVCLASS: dvc STATION: stnnm ON
                   LIB: l_dollar sitenm l_2points_3 efn l_2points_1 md l_2points_2 file_dvc */

/*  OU38  << 1 >>  cmdnm dvnm STARTED ON CLASSES: ou_class_lst DEVCLASS: dvc STATION: stnnm
                   WORKING ON ron : ouseqnb ouclass ousz */

/*  OU38  << 2 >>  cmdnm dvnm STARTED ON CLASSES: ou_class_lst DEVCLASS: dvc STATION: stnnm
                   WAITING FOR SITE OR DEVICE */

/*  OU38  << 3 >>  cmdnm dvnm STARTED ON CLASSES: ou_class_lst DEVCLASS: dvc STATION: stnnm
                   WAITING FOR DEVICE */

/*  OU39  << 01 >>  DLD ron :RDY= rdy_nb HLD= hold_nb WAIT= wait_nb OUT= out_nb LINES= oulnnb
                    PAGES= oupgnb */
```

```
/*  OU40  << 1 >>  cmdnm dvnm STARTED TOWARDS sitenm WORKING ON ron : ouseqnb ouclass ousz */

/*  OU40  << 2 >>  cmdnm dvnm STARTED TOWARDS sitenm WAITING FOR SITE OR DEVICE */

/*  OU40  << 3 >>  cmdnm dvnm STARTED TOWARDS sitenm WAITING FOR DEVICE */

/*  OU44  << 01 >>  cmdnm RON= ron star1 PRTY= ouprty star2 CLASS= ouclass star3 DEST= hstnm
                    point stnnm WHEN= when NAME= ounm COPIES= oucpnb star4 status
                    BANINF=(text1 text2 text3 text4 )

USM /*  OU45  << 01 >>  UNKNOWN GTWRITER DESTINATION */

USM /*  OU46  << 01 >>  UNABLE TO SEND THE PRINTING REQUEST TO GTWRITER */

USM /*  OU47  << 01 >>  ERROR IN COMMAND PARAMETERS : gr4 */

USM /*  OU48  << 01 >>  fullname TOO MANY SUBFILES gr4 */

/*  OU58  << 01 >>  ron : ouseqnb ouclass oustate */

/*  OU60  << 01 >>  l_32_a */

/*  OU61  << 01 >>  efn l_2_points sfn */

/*  OU62  << 01 >>  l_8_c oucpnb */

/*  OU63  << 1 >>  cmdnm dvnm MEANINGLESS: INVALID STATION AND/OR NO MATCHING CLASSES */

/*  OU63  << 2 >>  cmdnm dvnm MEANINGLESS: NO WRITER STARTED */

/*  OU63  << 3 >>  cmdnm dvnm MEANINGLESS: NOT STARTED */

/*  OU63  << 4 >>  cmdnm MEANINGLESS: dvnm ALREADY STARTED FOR stnnm ocl64 dvc */

/*  OU64  << 01 >> cmdnm oudvnm SPOOLING TERMINATED TOWARDS sitenm */

/*  OU64  << 2 >>  cmdnm oudvnm TERMINATED */

/*  OU64  << 3 >>  cmdnm oudvnm IN TERMINATION */

/*  OU64  << 4 >>  cmdnm oudvnm COMPLETED */

/*  OU66  << 01 >> cmdnm REJECTED: CONFLICT WITH PREVIOUS SOW ON dvnm */

/*  OU67  << 01 >>  cmdnm dvnm MEANINGLESS: WRITER IN TERMINATION */

/*  OU68  << 01 >>  cmdnm NOT ALLOWED FOR STATION stnnm */

/*  OU69  << 01 >>  cmdnm REJECTED: gr4 */
```

```
/*  OU70  << 1 >>  cmdnm SPOOL FAILED: gr4 */

/*  OU70  << 2 >>  cmdnm SPOOL REJECTED: NO ACTIVITY TOWARDS sitenm */

/*  OU70  << 3 >>  cmdnm SPOOL REJECTED: INVALID SITE sitenm */

/*  OU70  << 4 >>  cmdnm SPOOL REJECTED: NOT STARTED TOWARDS sitenm */

/*  OU71  << 1 >>  cmdnm dvnm SPOOLING STARTED TOWARDS sitenm FOR stnnm ou_class_lst dvc */

/*  OU71  << 2 >>  cmdnm dvnm SPOOLING TERMINATED TOWARDS sitenm FOR stnnm ou_class_lst dvc */

/*  OU72  << 1 >>  cmdnm SUCCESSFUL: dvnm STARTED ON FILE : dollar sitenm l_2points_3 efn
                   l_points sfn l_2points_1 md l_2points_2 file_dvc FOR stnnm ou_class_lst
                   dvc */

/*  OU72  << 2 >>  cmdnm SUCCESSFUL: dvnm STARTED ON LIB : dollar sitenm l_2points_3 efn
                   l_2points_1 md  l_2points_2 file_dvc FOR stnnm ou_class_lst dvc */

/*  OU72  << 3 >>  cmdnm SUCCESSFUL: dvnm STARTED FOR stnnm ou_class_lst dvc */

/*  OU73  << 1 >>  cmdnm SELCLASS UPDATED dvnm STARTED ON FILE: dollar sitenm l_2points_3 efn
                   l_points sfn l_2points_1 md l_2points_2 file_dvc FOR stnnm ou_class_lst
                   dvc */

/*  OU73  << 2 >>  cmdnm SELCLASS UPDATED dvnm STARTED ON LIB: dollar sitenm l_2points_3 efn
                   l_2points_1 md l_2points_2 file_dvc FOR stnnm ou_class_lst dvc */

/*  OU73  << 3 >>  cmdnm SELCLASS UPDATED dvnm STARTED FOR stnnm ou_class_lst dvc

/*  OU74  << 1 >>  cmdnm dvnm TERMINATED ON FILE: dollar sitenm l_2points_3 efn l_points sfn
                   l_2points_1 md l_2points_2 FOR stnnm ou_class_lst dvc */

/*  OU74  << 2 >>  cmdnm dvnm TERMINATED ON LIB: dollar sitenm l_2points_3 efn  l_2points_1 md
                   FOR stnnm ou_class_lst dvc */

/*  OU74  << 3 >>  cmdnm dvnm TERMINATED FOR stnnm ou_class_lst dvc */

/*  OU77  << 1 >>  cmdnm REJECTED: FILE: dollar sitenm l_2points_3 efn l_2points_1 md
                   l_2points_2 file_dvc unknown */

/*  OU77  << 2 >>  cmdnm REJECTED: INVALID RECSIZE */

/*  OU77  << 3 >>  cmdnm REJECTED: RBF OR DJP NOT PURCHASED */

/*  OU80  << 01 >>  cmdnm MEANINGLESS: NO OUTPUT oustate */

/*  OU81  << 1 >> cmdnm ron MEANINGLESS: NO OUTPUT oustate  */
/*  OU81  << 2 >> cmdnm ron NOT ALLOWED: NO OUTPUT (WILL BE)  oustate  */
```

```
/*  OU82  << 1 >> cmdnm ron MEANINGLESS: OUTPUT (OR JOB) DOES NOT EXIST  */
/*  OU82  << 2  >> cmdnm ron  MEANINGLESS:  JOB  DOES NOT EXIST  */

/*  OU83  << 1 >> cmdnm ron count OUTPUT(S) oustate  */
/*  OU83  << 2 >> cmdnm ron ALL OUTPUTS (WILL BE) oustate  */
/*  OU83  << 3 >> cmdnm ron ALL OUTPUTS AND JOB OUTPUT PARAMETERS oustate  */

/*  OU84  << 01 >> cmdnm ron REJECTED: INCORRECT OR UNSUFFICIENT ACCESS RIGHTS  */

/*  OU85  << 1 >> cmdnm ron:ouseqnb MEANINGLESS: OUTPUT CURRENTLY IN OUT  */
/*  OU85  << 2 >> cmdnm ron:ouseqnb MEANINGLESS: OUTPUT ALREADY IN OUT OR BEING
                  TERMINATED */
/*  OU85  << 3 >> cmdnm ron:ouseqnb NOT ALLOWED: OUTPUT BEING CREATED OR SCANNED */
/*  OU85  << 4 >> cmdnm ron:ouseqnb STARTED: ONLY COPIES NUMBER OR CLASS HAVE BEEN
                  oustate  */
/*  OU85  << 5 >> cmdnm ron:ouseqnb STARTED: OUTPUT  oustate  BUT SYSOUT CLEANING
                  NOT COMPLETED  */
/*  OU85  << 6 >> cmdnm ron:ouseqnb MEANINGLESS: OUTPUT ALREADY oustate  */
/*  OU85  << 7 >> cmdnm ron:ouseqnb WILL BE oustate BY WRITER OR GTP  */
/*  OU85  << 8 >> cmdnm ron:ouseqnb MEANINGLESS: OUTPUT TERMINATED (PRINTED OR
                  CANCELLED)  */

/*  OU86  << 01 >>  cmdnm ron : ouseqnb COMPLETED BUT NO WRITER MATCHES OUTPUT SELECTION
                    CRITERIA */

/*  OU88  << 01 >> cmdnm ron UNSUCCESSFUL: gr4  */

/*  OU90  << 01 >>  cmdnm ron : ounm   oustate  */

USM /*  OU95  << 1 >>  DEGRADED OUTPUT QUEUE RECOVERY: OUTPUT SEQUENCING LOST */

    /*  OU95  << 2 >>  OUTPUT QUEUE RECOVERY FAILURE: QUEUE INCOMPLETE */

    /*  OU95  << 3 >>  OUTPUT QUEUE RECOVERY FAILURE: QUEUE EMPTY */

    /*  OU95  << 4 >>  OUTPUT QUEUE RECOVERY FAILURE: QUEUE DISORDERED AND INCOMPLETE */

    /*  OU95  << 5 >>  OUTPUT QUEUE RECOVERED BUT INCONSISTENCIES DETECTED AT OUTPUT
                       RESEQUENCING */



/*  PM02  << 1  >> REPLY  SUCCESSFUL FOR  qid  */
/*  PM02  << 2  >> REPLY  MEANINGLESS FOR  qid  */
/*  PM02  << 3  >> REPLY  TRANSFER UNSUCCESSFUL FOR  qid  */
/*  PM02  << 4  >> REPLY  INTERNAL INCOHERENCE IN TREATMENT OF MESSAGE  qid  */

/*  PM03  << 1  >> DR  NO MORE PENDING REQUEST  */
/*  PM03  << 2  >> DR  NO PENDING REQUEST  */
/*  PM03  << 3  >> DR  INTERNAL INCOHERENCE IN TREATMENT  */
```

```
/*  PM04  << 1  >> DR  nb_requests  PENDING REQUEST  */
/*  PM04  << 2  >> DR  nb_requests  PENDING REQUESTS  */

/*  PM51  << 1  >> FUNCTION TEMPORARILY UNAVAILABLE : RETRY LATER. gr4  */
/*  PM51  << 2  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm OVERFLOW ON
                   COMMUNICATION  BUFFER  gr4  */
/*  PM51  << 3  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm TRUNCATED
                   COMMAND  gr4 */
/*  PM51  << 4  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm COMMAND CANNOT
                   BE RETRIEVED   gr4  */
/*  PM51  << 5  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm FUNCTION NOT
                   SUPPORTED BY REMOTE SYSTEM  gr4  */
/*  PM51  << 6  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm PMOS SESSION
                   NOT OPENED ON REMOTE SYSTEM  gr4  */
/*  PM51  << 7  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm gr4  */
/*  PM51  << 8  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm RETRY LATER
                   gr4  */
/*  PM51  << 9  >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm RAEH SERVER NOT
                   STARTED OR NOT READY gr4  */
/*  PM51  << 10 >> COMMAND  cmdnm  NOT SUBMITTED ON SYSTEM  sitenm VCAM SESSION
                   CLOSED BY NEH, RETRY LATER gr4  */
/*  PM51  << 11 >> NETWORK FAILURE DURING COMMAND EXECUTION ON SYSTEM
                   sitenm  gr4*/
/*  PM51  << 12 >> VCAM SESSION CLOSED BY NEH DURING COMMAND EXECUTION ON SYSTEM
                   sitenm RETRY LATER  gr4  */
/*  PM51  << 13 >> COMMAND  cmdnm  NOT EXECUTED ON SYSTEM  sitenm gr4  */
/*  PM51  << 14 >> WARNING: A RESPONSE HAS BEEN DAMAGED ON SYSTEM  sitenm
                   TRUNCATED RESPONSE    gr4  */
/*  PM51  << 15 >> WARNING: A RESPONSE HAS BEEN DAMAGED ON SYSTEM  sitenm THE
                   RESPONSE CANNOT BE RETRIEVED  gr4  */
/*  PM51  << 16 >> WARNING: A RESPONSE HAS BEEN DAMAGED ON SYSTEM  sitenm OVERFLOW
                   ON COMMUNICATION BUFFER  gr4  */
/*  PM51  << 17 >> WARNING: A MESSAGE HAS BEEN DAMAGED ON SYSTEM  sitenm TRUNCATED
                   MESSAGE gr4    */
/*  PM51  << 18 >> WARNING: A MESSAGE HAS BEEN DAMAGED ON SYSTEM  sitenm THE
                   MESSAGE CANNOT BE RETRIEVED  gr4  */
/*  PM51  << 19 >> WARNING: A MESSAGE HAS BEEN DAMAGED ON SYSTEM  sitenm OVERFLOW
                   ON COMMUNICATION BUFFER  gr4  */
/*  PM51  << 20 >> CONNECTIONS FOR REMOTE SYSTEM OPERATION CLOSED BY RAEH
                   SHUTDOWN  */



/*  SH01  << 1 >> cmdnm UNSUCCESSFUL : INIT OPTION NOT AVAILABLE  */
/*  SH01  << 2 >> cmdnm UNSUCCESSFUL : RESET OPTION NOT AVAILABLE  */
/*  SH01  << 3 >> cmdnm UNSUCCESSFUL : PWROFF OPTION NOT AVAILABLE  */
/*  SH01  << 4 >> cmdnm UNSUCCESSFUL : GCOS=1 REQUIRED */

/*  SH02  << 01 >> cmdnm UNSUCCESSFUL :  susp_nb  JOBS SUSPENDED  */
```

```
/*  SH03  << 1  >> cmdnm SUCCESSFUL  */
/*  SH03  << 2  >> cmdnm COMPLETED  */

/*  SH04  << 1  >> cmdnm ALREADY  FOR  ron  */
/*  SH04  << 2  >> cmdnm ALREADY  */

/*  SH05  << 1  >> cmdnm UNSUCCESSFUL : INVALID DATA  : SPR  spr  */
/*  SH05  << 2  >> cmdnm UNSUCCESSFUL : INVALID DATA  : XPR  dpr  */
/*  SH05  << 3  >> cmdnm UNSUCCESSFUL : INVALID DATA  : MULTLEV  multlev  */
/*  SH05  << 4  >> cmdnm UNSUCCESSFUL : INVALID DATA  : CLASS  jclass  */

USM /*  SH06  << 01 >> GCOS: NO MORE JOBS RUNNING  */

USM /*  SH07  << 01 >> GCOS: IDLE  */

/*  SH08  << 1  >> cmdnm UNSUCCESSFUL : NOT ALLOWED  SPR  spr  FOR  jclass  */
/*  SH08  << 2  >> cmdnm UNSUCCESSFUL : NOT ALLOWED  XPR  dpr  FOR  ron  */
/*  SH08  << 3  >> cmdnm UNSUCCESSFUL : NOT ALLOWED  MULTLEV  multlev  */
/*  SH08  << 4  >> cmdnm UNSUCCESSFUL : NOT ALLOWED  CLASS  jclass  FOR  ron  */

/*  SH09  << 1  >> cmdnm UNSUCCESSFUL : UNKNOWN JOB   ron  */
/*  SH09  << 2  >> cmdnm UNSUCCESSFUL : UNKNOWN JOB  */

/*  SH10  << 1  >> cmdnm UNSUCCESSFUL : CLASS  jclass  UNKNOWN BY SYSTEM  */
/*  SH10  << 2  >> cmdnm UNSUCCESSFUL : CLASS  jclass  NOT ALLOWED BY CATALOG  */

/*  SH11  << 1  >> cmdnm UNSUCCESSFUL  : TYPE OF  ron  MEANINGLESS WITH THE
                   COMMAND  */
/*  SH11  << 2  >> cmdnm UNSUCCESSFUL  : CLASS OF  ron  MEANINGLESS WITH THE
                   COMMAND  */
/*  SH11  << 3  >> cmdnm UNSUCCESSFUL  : STATE OF  ron  MEANINGLESS WITH THE
                   COMMAND  */
/*  SH11  << 4  >> cmdnm UNSUCCESSFUL  : ATTRIBUTES OF  ron  MEANINGLESS WITH THE
                   COMMAND */
/*  SH11  << 5  >> cmdnm UNSUCCESSFUL  : MULTPROCESS PG OF  ron  MEANINGLESS WITH
                   COMMAND */
/*  SH11  << 6  >> cmdnm UNSUCCESSFUL  : FORCE OPTION MEANINGLESS FOR  ron  */
/*  SH11  << 7  >> cmdnm UNSUCCESSFUL  */

/*  SH12  << 01 >> cmdnm UNSUCCESSFUL : CHECKING OF CATALOG NOT POSSIBLE  */

/*  SH13  << 1  >> cmdnm NOT ALLOWED  FOR  ron  */
/*  SH13  << 2  >> cmdnm NOT ALLOWED  FOR CLASS  jclass  */
/*  SH13  << 3  >> cmdnm NOT ALLOWED  FOR  ron  : SUBMITTER NOT AUTHORIZED  */
/*  SH13  << 4  >> cmdnm NOT ALLOWED  : SUBMITTER NOT AUTHORIZED  */
/*  SH13  << 5  >> cmdnm NOT ALLOWED  */

/*  SH14  << 1  >> ron jstate jnm   usernm   jclass  */
/*  SH14  << 2  >> ron . ssn jstate jnm   usernm   jclass   dimname   lm_name  */
```

```
/*  SH15  << 1  >> ron jstate jnm   usernm   jclass  SPR= spr  XPR= dpr  */
/*  SH15  << 2  >> ron . ssn jstate jnm   usernm   jclass  SPR= spr  XPR= dpr
                   dimname  lm_name  */

/*  SH16  << 1  >> LOAD: STARTED= started_nb / current_nb
                   BATCH= executing_nb + suspended_nb
                   IOF= iof_nb
                   SERVICE= service_nb  */
/*  SH16  << 2  >> LOAD: jclass1 = load_1 / multi_1  */
/*  SH16  << 3  >> LOAD: jclass1 = load_1 / multi_1
                         jclass2 = load_2 / multi_2  */
/*  SH16  << 4  >> LOAD: jclass1 = load_1 / multi_1
                         jclass2 = load_2 / multi_2
                         jclass3 = load_3 / multi_3  */
/*  SH16  << 5  >> LOAD: jclass1 = load_1 / multi_1
                         jclass2 = load_2 / multi_2
                         jclass3 = load_3 / multi_3
                         jclass4 = load_4 / multi_4  */
/*  SH16  << 6  >> LOAD: jclass1 = load_1 / multi_1
                         jclass2 = load_2 / multi_2
                         jclass3 = load_3 / multi_3
                         jclass4 = load_4 / multi_4
                         jclass5 = load_5 / multi_5  */

/*  SH17  << 1  >> CPU= cpu ELAPSED= elapsed  */
/*  SH17  << 2  >> SWAPPED ELAPSED=  elapsed  */

/*  SH18  << 1  >> SUCCESSFUL : TRACE ON  */
/*  SH18  << 2  >> SUCCESSFUL : TRACE OFF  */

/*  SH19  << 1  >>  WAITS FOR  FILE  efn  ON  volnm  */
/*  SH19  << 2  >>  WAITS FOR  VOLUME  volnm:dvtyp / dvattr labeltyp */
/*  SH19  << 3  >>  WAITS FOR  DEVICE  dvtyp  */
/*  SH19  << 4  >>  WAITS FOR  BEFORE JOURNAL  */
/*  SH19  << 5  >>  WAITS FOR  SHARED MEMORY SPACE  */
/*  SH19  << 6  >>  WAITS FOR  PROCESS  */
/*  SH19  << 7  >>  WAITS FOR  RESSOURCES IN DIMENSION  dimname  */
/*  SH19  << 8  >>  WAITS FOR  LOADING  */
/*  SH19  << 9  >>  WAITS FOR  SYSOUT SPACE  */
/*  SH19  << 10 >>  WAITS FOR  OPERATOR REPLY  repnb  */
/*  SH19  << 11 >>  WAITS FOR  VOLUME MOUNTING  */
/*  SH19  << 12 >>  WAITS FOR  FILE SPACE  */
/*  SH19  << 13 >>  WAITS FOR  TEMPORARY FILE SPACE ON DUAL RSDT  */
/*  SH19  << 14 >>  WAITS FOR  DUAL SHARING OF FILE  efn  ON  volnm  */
/*  SH19  << 15 >>  WAITS FOR  SITE  */
/*  SH19  << 16 >>  WAITS FOR  TELECOM SERVER  */
/*  SH19  << 17 >>  WAITS FOR  REMOTE RESOURCE  */
/*  SH19  << 18 >>  WAITS FOR  USER RECONNECTION  */
/*  SH19  << 19 >>  WAITS FOR  REMOTE FILE  */
/*  SH19  << 20 >>  WAITS FOR  REMOTE FILE  efn  ON  volnm  */
/*  SH19  << 21 >>  WAITS FOR  NON PREINITIALIZED LOAD MODULE RESSOURCE  */
/*  SH19  << 22 >>  WAITS FOR  LOADING PROCESS  */
```

```
/*  SH20  << 1  >> jclass1 = load_1  */
/*  SH20  << 2  >> jclass1 = load_1 jclass2 = load_2  */
/*  SH20  << 3  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3  */
/*  SH20  << 4  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3 jclass4 = load_4  */
/*  SH20  << 5  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3 jclass4 = load_4
                   jclass5 = load_5  */
/*  SH20  << 6  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3 jclass4 = load_4
                   jclass5 = load_5 jclass6 = load_6  */
/*  SH20  << 7  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3 jclass4 = load_4
                   jclass5 = load_5 jclass6 = load_6
                   jclass7 = load_7  */
/*  SH20  << 8  >> jclass1 = load_1 jclass2 = load_2
                   jclass3 = load_3 jclass4 = load_4
                   jclass5 = load_5 jclass6 = load_6
                   jclass7 = load_7 jclass8 = load_8  */

/*  SH21  << 01 >> IN  = in_nb HOLD= hold_nb SCH = sch_nb EX  = ex_nb SUSP=
                   susp_nb OUT = out_nb IDLE= idle_nb  */

/*  SH22  << 1  >> jclass START LOAD= load / multi OTHER= sch_load
                   + nosch_load PRTY=( spr , dpr )  MDMULT= yorn  */
/*  SH22  << 2  >> jclass NSTART LOAD= load / multi OTHER= sch_load
                   + nosch_load PRTY=( spr , dpr )  MDMULT= yorn  */

USM /*  SH23  << 1  >> cmdnm ron   HELD  */
USM /*  SH23  << 2  >> cmdnm ron . ssn   SUSPENDED  */
USM /*  SH23  << 3  >> cmdnm ron   FORCED BY OPERATOR  */

/*  SH24  << 1  >> cmdnm ron   RELEASED  */
/*  SH24  << 2  >> cmdnm ron . ssn   RELEASED  */

/*  SH25  << 01 >> cmdnm  STRONG REQUIRED FOR  ron  : HOLD COUNT= holdcnt  */

/*  SH26  << 01 >> NO JOB PROCESSED BY THE COMMAND :  cmdnm  */

/*  SH27  << 01 >> cmdnm UNSUCCESSFUL : INVALID PRIORITY FOR  ron  IN CLASS
                   jclass  */

/*  SH28  << 01 >> NO JOBS SELECTED BY DISPLAY  */

/*  SH29  << 01 >> msg  */
```

```
/*  SH30  << 1  >> ron . ssn MODIFIED : CLASS= jclass SPR= spr
                   XPR= dpr SW= swtchlst  */
/*  SH30  << 2  >> ron . ssn MODIFIED : CLASS= jclass SPR= spr
                   XPR= dpr  */
/*  SH30  << 3  >> ron . ssn MODIFIED : CLASS= jclass SPR= spr
                   SW= swtchlst  */
/*  SH30  << 4  >> ron . ssn MODIFIED : CLASS= jclass SPR= spr  */
/*  SH30  << 5  >> ron . ssn MODIFIED : CLASS= jclass XPR= dpr
                   SW= swtchlst  */
/*  SH30  << 6  >> ron . ssn MODIFIED : CLASS= jclass XPR= dpr  */
/*  SH30  << 7  >> ron . ssn MODIFIED : CLASS= jclass
                   SW= swtchlst  */
/*  SH30  << 8  >> ron . ssn MODIFIED : CLASS= jclass  */
/*  SH30  << 9  >> ron MODIFIED : CLASS= jclass SPR= spr XPR= dpr
                   SW= swtchlst  */
/*  SH30  << 10 >> ron MODIFIED : CLASS= jclass SPR= spr XPR= dpr  */
/*  SH30  << 11 >> ron MODIFIED : CLASS= jclass SPR= spr
                   SW= swtchlst  */
/*  SH30  << 12 >> ron MODIFIED : CLASS= jclass SPR= spr  */
/*  SH30  << 13 >> ron MODIFIED : CLASS= jclass XPR= dpr
                   SW= swtchlst  */
/*  SH30  << 14 >> ron MODIFIED : CLASS= jclass XPR= dpr  */
/*  SH30  << 15 >> ron MODIFIED : CLASS= jclass SW= swtchlst  */
/*  SH30  << 16 >> ron MODIFIED : CLASS= jclass  */

/*  SH31  << 1  >> cmdnm UNSUCCESSFUL :  INVALID RANGES OF CLASSES  */
/*  SH31  << 2  >> cmdnm UNSUCCESSFUL :  RANGES OF CLASSES OR ALL MUST
                   BE SPECIFIED  */
/*  SH31  << 3  >> cmdnm UNSUCCESSFUL :  CLASS  jclass  CANNOT BE A
                   CLASS GROUP  */
/*  SH31  << 4  >> cmdnm UNSUCCESSFUL :  CLASS  jclass  CONNECTED
                   TO ANOTHER CLASS GROUP  */
/*  SH31  << 5  >> cmdnm UNSUCCESSFUL :  CLASS GROUP AND ITS SON
                   CLASS CANNOT BE THE SAME  */
/*  SH31  << 6  >> cmdnm UNSUCCESSFUL :  TOO MUCH ERROR MESSAGES TO
                   BE DISPLAYED  */
/*  SH31  << 7  >> cmdnm UNSUCCESSFUL :  CLASS  jclass  IS A CLASS
                   GROUP  */

/*  SH32  << 1  >> class   GROUP:  f_jclass   CLASS:  jclass1
                   jclass2   jclass3   jclass4 jclass5   jclass6
                   jclass7   jclass8   jclass9   jclass10
                   jclass11   jclass12   jclass13   jclass14  */
/*  SH32  << 2  >> CLASS:  jclass1   jclass2   jclass3   jclass4
                   jclass5   jclass6   jclass7   jclass8   jclass9
                   jclass10   jclass11   jclass12   jclass13
                   jclass14   jclass15   jclass16   jclass17
                   jclass18  */

/*  SH33  << 1  >> ron . ssn  SUSPENDED BY SYSTEM  */
/*  SH33  << 2  >> ron . ssn  RELEASED BY SYSTEM  */
```

```
/*  SH34  << 1  >> cmdnm NOT ALLOWED FOR  jclass  (CAN NOT BE DELETED)  */
/*  SH34  << 2  >> cmdnm NOT ALLOWED FOR  jclass  (ALREADY EXISTS)  */
/*  SH34  << 3  >> cmdnm UNSUCCESSFUL : INVALID RANGE OF CLASSES  */
/*  SH34  << 4  >> cmdnm UNSUCCESSFUL : NOT ALLOWED CLASSLIST  */
/*  SH34  << 5  >> cmdnm ABORTED : MAX NB OF CLASSES IS REACHED  */
/*  SH34  << 6  >> cmdnm COMPLETED : NO CLASS SELECTED BY COMMAND  */

/*  SH35  << 1  >> CLASS :  */
/*  SH35  << 2  >> GROUP :  */

/*  SH40  << 1  >> jclass START EX+SUSP=load SCH=sch_load OTHER=other_load
                   MULTI=multi MDMULT=yorn  */
/*  SH40  << 2  >> jclass NSTART EX+SUSP=load SCH=sch_load OTHER=other_load
                   MULTI=multi MDMULT=yorn  */

/*  SH41  << 1  >> SPR=spr SPRLIM=sprlim XPR=dpr XPRLIM=xprlim  */
/*  SH41  << 2  >> SPR=spr SPRLIM=sprlim XPR=dpr XPRLIM=xprlim ELAPTIME=elaptime  */
/*  SH41  << 3  >> SPR=spr SPRLIM=sprlim XPR=dpr XPRLIM=xprlim CPTIME=cptime  */
/*  SH41  << 4  >> SPR=spr SPRLIM=sprlim XPR=dpr XPRLIM=xprlim ELAPTIME=elaptime
                   CPTIME=cptime  */

/*  SH42  << 1  >> WAITS FOR TEMPORARY FILE SPACE ON VOLSET volsetnm  */
/*  SH42  << 2  >> WAITS FOR FILE SPACE ON VOLSET volsetnm  */
/*  SH42  << 3  >> WAITS FOR FILE RESTORATION:efn  */
/*  SH42  << 4  >> WAITS FOR FILE SPACE ON DISK md  */
/*  SH42  << 5  >> WAITS FOR SPACE ON VOLSET volsetnm  */

/*  SH43  << 1  >> cmdnm UNSUCCESSFUL: CLASS jclass CANNOT BE CONNECTED TO A
                   JOBCLASS GROUP  */

/*  SH44  << 1  >> NON TEMPORARY PASSING FILE EXISTS FOR ron, HJ CAN LEAD TO A DEADLOCK  */

/*  SH45  << 1  >> cmdnm ENDSTEP APPLIED ON count EXECUTING JOB(S) WILL BE EFFECTIVE AS SOON
                   AS JOBSTATE PERMITS  */
/*  SH45  << 2  >> cmdnm APPLIED ON count EXECUTING JOB(S) WILL BE EFFECTIVE AS SOON
                   AS JOBSTATE PERMITS  */
/*  SH45  << 3  >> cmdnm RECEIVED - APPLICATION IN PROGRESS  */

/*  SH46  << 1  >> cmdnm UNSUCCESSFUL: UNKNOWN JOB ron  */
/*  SH46  << 2  >> cmdnm UNSUCCESSFUL: STATE OF ron MEANINGLESS WITH THE COMMAND  */
/*  SH46  << 3  >> cmdnm UNSUCCESSFUL FOR ron gr4  */
/*  SH46  << 4  >> cmdnm NOT ALLOWED FOR ron: SUBMITTER NOT AUTHORIZED  */
/*  SH46  << 5  >> cmdnm UNSUCCESSFUL FOR ron: NO FILE ASSIGNED  */
/*  SH46  << 6  >> cmdnm UNSUCCESSFUL FOR efn: FILE NOT ASSIGNED  */
/*  SH46  << 7  >> cmdnm UNSUCCESSFUL FOR efn gr4  */


/*  SH50  << 01  >> EFN= efn MD= ( md1 / md2 / md3 / md4 / md5 / md6 / md7 / md8 /
                     md9 / md10 ) DVC= dvc FILESTAT= filestat  */
```

```
/*  SH51  << 01  >> head ron1 ron2 ron3 ron4 ron5 ron6 ron7 ron8  */

/*  SH52  << 1  >>  RON= ron  */
/*  SH52  << 2  >>  IFN= ifn PMD= pmd  */

/*  SH53  << 01 >>  ------------------------------------------------------------  */


/*  TF05  << 1  >> START LOG BY SERVER  svr_name  ON OPERATOR REQUEST EFN:  efn */
/*  TF05  << 2  >> START LOG BY SERVER  svr_name  ON SEVERITY EFN:  efn  */
/*  TF05  << 3  >> START LOG BY SERVER  svr_name  ON ABORT EFN:  efn  */
/*  TF05  << 4  >> LOG OF SERVER  svr_name  ALREADY ACTIVE EFN:  efn  */

/*  TF06  << 1  >> LOG TERMINATED BY SERVER  svr_name  ON OPERATOR REQUEST
                   EFN:  efn  */
/*  TF06  << 2  >> LOG TERMINATED BY SERVER  svr_name  ON ITS TERMINATION EFN:
                   efn  */
/*  TF06  << 3  >> LOG OF SERVER  svr_name  ALREADY INACTIVE. LAST EFN (IF ANY) :
                   efn  */

/*  TF07  << 1  >> TRACE WARNING ( error_code ) FOR SERVER  svr_name   gr4
                   INCONSISTENT TRACE USE. SERVER NOT DAMAGED PLEASE CALL YOUR
                   FIELD ENGINEER   */
/*  TF07  << 2  >> TRACE WARNING ( error_code ) FOR SERVER  svr_name   gr4 FILE
                   MANAGEMENT INCIDENT   */
/*  TF07  << 3  >> TRACE WARNING ( error_code ) FOR SERVER  svr_name   gr4 MEMORY
                   MANAGEMENT INCIDENT   */
/*  TF07  << 4  >> TRACE WARNING ( error_code ) FOR SERVER  svr_name   gr4
                   UNEXPECTED INCIDENT.   SERVER NOT DAMAGED PLEASE CALL YOUR
                   FIELD ENGINEER   */

/*  TF50  << 1  >> DBSVR  svr_name  MD  COMPLETED  */
/*  TF50  << 2  >> DBSVR  svr_name  MD  NO OBJECT SELECTED  */
/*  TF50  << 3  >> DBSVR  svr_name  MD  NO SUBDOMAIN SELECTED  */
/*  TF50  << 4  >> DBSVR  svr_name  MD  UNKNOWN SVR_NAME  */
/*  TF50  << 5  >> DBSVR  svr_name  MD  IS THE COMMAND TO USE. (UNKNOWN REQUESTED
                   ACTION)  */
/*  TF50  << 6  >> DBSVR  svr_name  MD  IS THE COMMAND TO USE. (UNKNOWN REQUESTED
                   TARGET)  */
/*  TF50  << 7  >> DBSVR  svr_name  MD  REFUSED FOR AT LEAST ONE OBJECT :
                   POS+VAL > LENGTH */
/*  TF50  << 8  >> DBSVR  svr_name  MD  IMPOSSIBLE (NO TABLES) : SVR_NAME NEVER
                   STARTED  */
/*  TF50  << 9  >> DBSVR  svr_name  MD  LOG IMPOSSIBLE : SVR_NAME NOT ACTIVE  */
/*  TF50  << 10 >> DBSVR  svr_name  MD  NO DOMAIN SELECTED  */
/*  TF50  << 11 >> DBSVR  svr_name  MD  MAY BE PERFORMED. PLEASE CHECK WITH DBSVR
                   [D]  */
```

```
/*  TF51  << 1  >> DBSVR  svr_name  D  COMPLETED  */
/*  TF51  << 2  >> DBSVR  svr_name  D  NO OBJECT SELECTED  */
/*  TF51  << 3  >> DBSVR  svr_name  D  NO SUBDOMAIN SELECTED  */
/*  TF51  << 4  >> DBSVR  svr_name  D  UNKNOWN SVR_NAME  */
/*  TF51  << 5  >> DBSVR  svr_name  D  IS THE COMMAND TO USE. (UNKNOWN REQUESTED
                   ACTION) */
/*  TF51  << 6  >> DBSVR  svr_name  D  IS THE COMMAND TO USE. (UNKNOWN REQUESTED
                   TARGET) */
/*  TF51  << 7  >> DBSVR  svr_name  D  WARNING FOR AT LEAST ONE OBJECT : POS+VAL >
                   LENGTH */
/*  TF51  << 8  >> DBSVR  svr_name  D  IMPOSSIBLE (NO TABLES) : SVR_NAME NEVER
                   STARTED  */
/*  TF51  << 9  >> DBSVR  svr_name  D  LOG IMPOSSIBLE : SVR_NAME NOT ACTIVE  */
/*  TF51  << 10 >> DBSVR  svr_name  D  NO DOMAIN SELECTED  */
/*  TF51  << 11 >> DBSVR  svr_name  D  COMPLETED WITH ERROR  */

/*  TF52  << 1  >> LOG INACTIVE.  LAST EFN (IF ANY):  efn CATALOGUE :   fs_
                   RESIDENT  :  rd_   SUFFIX    :   sfx_ MEDIA     :   md_  */
/*  TF52  << 2  >> LOG ACTIVE    EFN:  efn CATALOGUE :   fs_ RESIDENT  :   rd_
                   SUFFIX    :    sfx_ MEDIA     :   md_  */
/*  TF52  << 3  >> LOG ACTIVE (ON SEVERITY)    EFN:  efn CATALOGUE :   fs_
                   RESIDENT  :   rd_   SUFFIX    :   sfx_ MEDIA     :   md_  */
/*  TF52  << 4  >> LOG ACTIVE (ON ABORT)    EFN:  efn CATALOGUE :   fs_
                   RESIDENT  :   rd_   SUFFIX    :   sfx_ MEDIA     :   md_  */

/*  TF53  << 01 >> OBJECT ID LENGTH FILTER ADDRESS POSITION VALUE  */

/*  TF54  << 1  >> obj_name obj_id obj_length ON addr position value  */
/*  TF54  << 2  >> obj_name obj_id obj_length OFF addr position value  */
/*  TF54  << 3  >> obj_name obj_id obj_length ADDR addr position value  */
/*  TF54  << 4  >> obj_name obj_id obj_length POSVAL addr position value  */
/*  TF54  << 5  >> obj_name obj_id obj_length AND addr position value  */
/*  TF54  << 6  >> obj_name obj_id obj_length OR addr position value  */
/*  TF54  << 7  >> obj_name obj_id obj_length flt_obj addr position value  */

/*  TF55  << 01 >> SUBDOM IN DOMAIN ID SICID LVL SEV TOL SW0 SW1 SW2  */

/*  TF56  << 01 >> subd_name dom_name subd_id sicid level severity tolerance sw0
                   sw1 sw2 */

/*  TF57  << 01 >> DOMAIN ID DEBUG  */

/*  TF58  << 1  >> dom_name id 1  */
/*  TF58  << 2  >> dom_name id 0  */
/*  TF58  << 3  >> dom_name id dom_state  */
```

```
/*  TM03  << 01 >> MDTIME ACCEPTED  */

/*  TM04  << 01 >> MDTIME REJECTED: PARAMETER VALUE OUT OF RANGE  */

/*  TM05  << 01 >> MDTIME REJECTED: JOURNAL AFTER IS RUNNING  */

/*  TM06  << 01 >> MDTIME REJECTED: SYNTAX ERROR  */

/*  TM08  << 01 >> NEW TIMEDEV IS  gcos_tdev , DO YOU AGREE? (Y OR N)  */

/*  TM09  << 01 >> MDTIME IGNORED  */

/*  TM10  << 01 >> MDTIME REJECTED: UNABLE TO ACCESS TO SP. RETRY LATER  */

/*  TM11  << 01 >> MDTIME <SYNC> MEANINGLESS  */

USM /*  TM12  << 01 >> SP TIMEDEV HAS BEEN UPDATED FROM OTHER SYSTEM  */

USM /*  TM13  << 01 >> GCOS7 TIMEDEV IS  gcos_tdev , SP TIMEDEV IS spos_tdev  */

USM /*  TM14  << 01 >> UPDATE TIMEDEV OF CURRENT SYSTEM AS SOON AS POSSIBLE  */

/*  TM15  << 01 >> MDTIME REJECTED: RUNNING OR WAITING FOR REPLY */

/*  TM16  << 01 >> MDTIME MEANINGLESS  */

/*  TM17  << 01 >> INVALID REPLY  */

/*  TM18  << 01 >> <Journal After> MAY BE IN PREDATING MODE TILL time_date  */


/*  TX00  << 01 >>  */

/*  TX01  << 01 >> REQUESTED TDS  tds  NOT AVAILABLE  */

/*  TX02  << 01 >> message  */


/*  TX04  << 01 >> POOL USED (KB)   =  poolused POOL SIZE (KB)   =  poolsize
                   PSEUDO BUFFERS   =  pseudobuf  */

/*  TX05  << 01 >> INIT.SIMU.COUNT  =  init_simu CUR.SIMU.COUNT   =  cur_simu
                   ACC.SESS.ALLOC   =  sess_alloc ACC.SESS.REJEC   =  sess_reject
                   USED TX COUNT    =  txnb TX ABORT. COUNT  =  txabtnb
                   USED TPR COUNT   =  tprnb  TPR ABORT COUNT  =  tprabtnb
                   COMMIT COUNT     =  cmtnb  DIALOG COUNT     =  dialnb  */
```

```
/*  TX06  << 01 >> TPR ELAPSED TIME =  tprelps TPR CPU TIME     =  tprcpu
                   DEADLOCK COUNT   =  ddlcknb NON CONCUR WAIT  =  nconcnb
                   TABOV ABT COUNT  =  tabovnb WDNAV ABT COUNT  =  wdnavnb
                   LGWAITABT COUNT  =  lgwaitnb DIRTY READ ABORT =  dirtabtnb
                   BUFOVABT COUNT   =  bufovnb   SERIALIZATION    =  serialnb  */

/*  TX07  << 01 >> MAX TM SES       =  mxtmnb CUR TM SES       =  curtmnb
                   MAX XCP1 SES     =  mxxcp1nb CUR XCP1 SES     =  curxcp1nb
                   MAX XCP2 SES     =  mxxcp2nb CUR XCP2 SES     =  curxcp2nb
                   MAX VIRT SES     =  dummax CUR VIRT SES     =  virtsesnb
                   PMOS COR COUNT   =  pmossesnb  */

/*  TX08  << 01 >> authcode1 authcode2 authcode3 authcode4 authcode5 authcode6
                   authcode7 authcode8 authcode9 authcode10 authcode11 authcode12
                   authcode13 authcode14 authcode15 authcode16 authcode17
                   authcode18 authcode19 authcode20 authcode21 authcode22
                   authcode23 authcode24 authcode25 authcode26 authcode27
                   authcode28 authcode29 authcode30 authcode31 authcode32  */

/*  TX09  << 01 >> MAX.SIMU.COUNT    =  init_simu CUR.SIMU.COUNT    =  cur_simu
                   FROZEN SIMU COUNT =  frz_simu XCP2 SIMU COUNT   =  xcp2_simu
                   NXCP2 SIMU COUNT  =  nxcp2_simu  */

/*  TX10  << 01 >> SM LIBRARY 1 :  smlib1 SM LIBRARY 2 :  smlib2 SM LIBRARY 3 :
                   smlib3  */

/*  TX11  << 01 >> XCP2 SERVICE USED               :  xcp2used PRIORITY  :  prty
                   TRANSACTION STORAGE SIZE        :  txstosz ACCOUNTING   :
                   account
                   FORM                            :  form  */

/*  TX12  << 01 >> MESSAGE        :  cmes   FIRST ASSIGNED TPR      :       cname
                   CLASS          :  lclass IMPLICIT COMMITMENT     :       impcom
                   AUTOMATIC UNMAPPING             :  unmap  */

/*  TX13  << 1  >> DISPLAY _TX  :  txnm   FILE SECURITY OPTION : SUPPRESS BEFORE
                   JOURNAL  */
/*  TX13  << 2  >> DISPLAY _TX  :  txnm   FILE SECURITY OPTION : SUPPRESS DEFERRED
                   UPDATES */
/*  TX13  << 3  >> DISPLAY _TX  :  txnm   USE DEFERRED UPDATES   */
/*  TX13  << 4  >> DISPLAY _TX  :  txnm   USE DEFERRED UPDATES EXCEPT FOR :  */
/*  TX13  << 5  >> DISPLAY _TX  :  txnm   SUPPRESS CONCURRENT ACCESS CONTROL
                   FOR :  */
/*  TX13  << 6  >> DISPLAY _TX  :  txnm   SHARED READ FOR :  */
/*  TX13  << 7  >> DISPLAY _TX  :  txnm   TX MANUALLY NON CONCURENT WITH  */
/*  TX13  << 8  >> DISPLAY _TX  :  txnm   TX NON CONCURRENT WITH :  */
/*  TX13  << 9  >> DISPLAY _TX  :  txnm   NO CLAUSE SPECIFIED  */
/*  TX13  << 10 >> DISPLAY _TX  :  txnm   TX MANUALLY NON CONCURENT WITH
                   ALL TX  */
/*  TX13  << 11 >> DISPLAY _TX  :  txnm   TX NON CONCURRENT WITH ALL  */
/*  TX13  << 12 >> DISPLAY _TX  :  txnm   LIST OF AUTHORITY CODES  */
```

```
/*  TX14   << 01 >> txnm1 txnm2 txnm3 txnm4 txnm5 txnm6  */

/*  TX15   << 01 >> ifn1 ifn2 ifn3 ifn4 ifn5 ifn6  */

/*  TX16   << 01 >> LOCKED TRANSACTION          : lock     FOR DEBUG TX        :
                    validate
                    FOR INQUIRY TX              : inquiry  HIDDEN (IN MENU) TX :
                    hidden  */

USM /*  TX17  << 1  >>    IFN    OPEN  MONITORED    PMD  */
USM /*  TX17  << 2  >> ifn open monitored pmd  */

/*  TX18  << 1  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST OF OPENED FILES       -----
                    ----------------------------------------     */

/*  TX18  << 2  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST OF CLOSED FILES       -----
                    ----------------------------------------     */

/*  TX18  << 3  >>  ----------------------------------------     -----
                    dt_tm      ---- -----          LIST OF FILES         -----
                    ----------------------------------------     */

/*  TX19 << 01 >> XCP2 HEURISTIC                   :  xcp2hrs NO DEFERRED
                  RESYNCHRONIZATION: nodfrsc NOT RESTARTABLE COMMITMENT (XCP2):
                  norstrt */

USM /*  TX20 << 01 >> SWAP_FILE  ACT  LOGGED_CNTXT  OCCUPANCY   ALLOC_FACTOR  */

USM /*  TX21 << 01 >> swapifn active ctxnb occup % alfactor %  */

/*  TX22  << 01 >> cornm1 cornm2 cornm3 cornm4 cornm5  */

/*  TX23  << 01 >> CORRESPONDENT     ADDRESS       TX_COUNT    TPR_COUNT  STATUS
                    TX_NM  */

/*  TX24  << 01 >> cornm coraddr txnb tprnb status txnm  */

/*  TX25  << 1  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST OF CORRESPONDENTS     -----
                    ----------------------------------------     */
/*  TX25  << 2  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST  OF TRANSACTIONS      -----
                    ----------------------------------------     */
/*  TX25  << 3  >>  ----------------------------------------     -----
                    dt_tm      ---- --  CHARACTERISTICS OF THE TRANSACTION  --
                    ----------------------------------------     */
```

```
/*  TX25  << 4  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST  OF POOLS             -----
                    ----------------------------------------     */
/*  TX25  << 5  >>  ----------------------------------------     -----
                    dt_tm      ---- -----     LIST  OF SPAWNEES          -----
                    ----------------------------------------     */
/*  TX25  << 6  >>  ----------------------------------------     -----
                    dt_tm      ---- --  LIST OF SPAWNNES WITH QUEUE LENGTHS --
                    ----------------------------------------     */

/*  TX26  << 1  >> CORRESPONDENT     ADDRESS      COR_BACKUP    */
/*  TX26  << 2  >> cornm coraddr corbck  */

/*  TX27  << 1  >> CORRESPONDENT     ADDRESS      NB_OF_POOLS    */
/*  TX27  << 2  >> cornm coraddr poolnb  */

/*  TX28  << 01 >> STATE :  state  / TYPE :  cortyp  / LIST :  list  / OPTION :
                   option  */

/*  TX29  << 1  >> CORRESPONDENT   ADDRESS   MXALCSES  CUMALCSES  REJALCSES  */
/*  TX29  << 2  >> cornm coraddr mxalses cumalses rejalses  */

/*  TX30  << 1  >>  ----------------------------------------     -----
                    dt_tm      ---- -----   GENERAL TDS STATISTICS       -----
                    ----------------------------------------     */
/*  TX30  << 2  >>  ----------------------------------------     -----
                    dt_tm      ---- -----      LIST OF SWAP FILES        -----
                    ----------------------------------------     */
/*  TX30  << 3  >>  ----------------------------------------     -----
                    dt_tm      ---- -----  CURRENT SIMULTANEITY LEVELS   -----
                    ----------------------------------------     */
/*  TX30  << 4  >>  ----------------------------------------     -----
                    dt_tm      ---- ----- CURRENT SEARCH RULES OF SM LIB -----
                    ----------------------------------------     */
/*  TX30  << 5  >>  ----------------------------------------     -----
                    dt_tm      ---- -----  CHARACTERISTICS OF THE POOL   -----
                    ----------------------------------------     */

/*  TX31  << 01 >> poolnm1 poolnm2 poolnm3 poolnm4 poolnm5 poolnm6  */

/*  TX32  << 1  >>  POOL    MAXSESNB  MINWINSC  MINWINTG  AUTOACT  DRAINSC
                    DRAINTG MAXSYNC   */
/*  TX32  << 2  >>  poolnm maxsesnb minwinsc minwintg autoact drainsc draintg
                    maxsync  */

/*  TX33  << 1  >> CORRESPONDENT     ADDRESS      COR_BACKUP    PRIM  ACTIVE
                   INITWK  */
/*  TX33  << 2  >> cornm coraddr corbck prim activ initwk  */
```

```
/*  TX34  << 1  >> CORRESPONDENT     ADDRESS      COR_BACKUP    PRIM  PRL  WIN
                   SYNLVL  */
/*  TX34  << 2  >> cornm coraddr corbck prim prl win synclv  */

/*  TX35  << 1  >> POOL    CURSESNB  CURWINSC  CURWINTG  TRSTSSNB  */
/*  TX35  << 2  >> poolnm cursesnb curwinsc curwintg trstssnb  */

/*  TX36  << 1  >> ATTRIB.   ACTSESNB  FRZSESNB  TRSTSESNB  */
/*  TX36  << 2  >> attrib actsesnb frzsesnb trstssnb  */

/*  TX38  << 1  >> CORRESPONDENT   HIGH_SPW_CNT   MED_SPW_NB      LOW_SPW_NB
                   TIMER_SPW_NB  */
/*  TX38  << 2  >> cornm highspnb medspnb lowspnb timspnb  */

USM /*  TX50  << 1  >> TDS :  tds  COLD RESTART IS PERFORMED  */
USM /*  TX50  << 2  >> TDS :  tds  WARM RESTART IS PERFORMED  */
USM /*  TX50  << 3  >> TDS :  tds  STARTED; YOU ARE MASTER TERMINAL OPERATOR  */
USM /*  TX50  << 4  >> TDS :  tds  STARTED WITH MASTER MAILBOX : mstmbx  */

USM /*  TX51  << 1  >> cornm  DISCONNECTED FROM  coraddr  */
USM /*  TX51  << 2  >> cornm  DISCONNECTED FROM TDS  tds  */

/*  TX52  << 1  >> cmdnm  COMMAND NOT PERFORMED, UNKNOWN TX :  txnm  */
/*  TX52  << 2  >> cmdnm  COMMAND NOT PERFORMED, INVALID TX :  txnm  */

USM /*  TX53  << 01 >> TDS :  tds  SHUTDOWN  */

/*  TX54  << 01 >> cmdnm  COMMAND COMPLETED  */

/*  TX55  << 01 >> cmdnm  COMMAND NOT PERFORMED  rc  */

/*  TX56  << 1  >> cmdnm  WRONG TYPE OF CORRESPONDENT  */
/*  TX56  << 2  >> cmdnm  UNKNOWN OR NO MATCHING CORRESPONDENT :  cornm  */

/*  TX57  << 01 >> cmdnm  SCROLLING UP MORE? (ENTER Y/N)   */

USM /*  TX58  << 01 >> OVERFLOW ON PMOS COMMAND/RESPONSES QUEUE  */

USM /*  TX59  << 01 >> ONLY ONE PROGRAMMED OPERATOR ALLOWED TO CONTROL TDS  */

USM /*  TX60  << 01 >>  COMMAND  cmdnm  IGNORED; RETRY LATER  */

/*  TX61  << 01 >> SM LIBRARY NOT FOUND; ISSUE DISPLAY_TDS TO CHECK  */

USM /*  TX62  << 01 >> SWAP CONTEXT MAXIMUM REACHED  */

/*  TX63  << 1  >> cmdnm  :  ifn  OPENED  */
/*  TX63  << 2  >> cmdnm  :  ifn  CLOSED  */

/*  TX64  << 1  >> cmdnm  :  ifn  NOT OPENED    gr4  */
/*  TX64  << 2  >> cmdnm  :  ifn  NOT CLOSED    gr4  */

/*  TX65  << 01 >>  UNKNOWN MASTER COMMAND :  cmdnm  */
```

```
/*  TX66  << 1  >> cmdnm  COMMAND NOT SUPPORTED FOR PROGRAMMED OPERATOR  */
/*  TX66  << 2  >> cmdnm  COMMAND NOT PERFORMED, SHUTDOWN IN PROGRESS  */

/*  TX67  << 1  >> cmdnm    tobjnm  LOADED INTO MAIN MEMORY  */
/*  TX67  << 2  >> cmdnm    tobjnm  UNLOADED FROM MAIN MEMORY  */

/*  TX68  << 1  >> cornm  :   UNRECIO ON CI NUMBER :  cinb  FOR  efn  */
/*  TX68  << 2  >> cornm  :   UNRECIO ON BEFORE JOURNAL  */

/*  TX69  << 1  >> UNKNOWN TX :  txnm  , SPAWN CANNOT BE STARTED FOR  cornm  */
/*  TX69  << 2  >> ACCESS DENIED TO TX :  txnm  , SPAWN CANNOT BE STARTED FOR
                   cornm  */

/*  TX70  << 01 >> poolnb  POOL(S) OPENED TOWARD  cornm  */

/*  TX71  << 1  >>  POOL  poolnm  OPENED BY  cornm  */
/*  TX71  << 2  >>  POOL  poolnm  MODIFIED BY  cornm  */
/*  TX71  << 3  >>  POOL  poolnm  CLOSED BY  cornm  */

/*  TX72  << 1  >> cmdnm COMMAND NOT PERFORMED FOR poolnm TDS REASON :  tdsreason  */
/*  TX72  << 2  >> cmdnm COMMAND NOT PERFORMED FOR poolnm NETGEN REASON :
                   netgen_rc  */
/*  TX72  << 3  >> cmdnm COMMAND NOT PERFORMED FOR poolnm PPC REASON : ppc_rc  /
                   ppcreason  */

/*  TX73  << 1  >> cmdnm  COMMAND NOT PERFORMED  TDS REASON :  tdsreason  */
/*  TX73  << 2  >> cmdnm  COMMAND NOT PERFORMED  NETGEN REASON :  netgen_rc  */
/*  TX73  << 3  >> cmdnm  COMMAND NOT PERFORMED  PPC REASON :  ppc_rc  /
                   ppcreason  */

/*  TX74  << 01 >> cmdnm  COMMAND NOT PERFORMED FOR  cornm  TDS REASON :
                   tdscode  */

USM /*  TX75  << 1  >> cmdnm  LIMIT IS REACHED, ONLY  sessnb  CREATED  */
USM /*  TX75  << 2  >> cmdnm  LIMIT IS REACHED, ONLY  sessnb  DELETED  */

/*  TX76  << 1  >> TDS :  tds  SUPERVISION ACTIVATED  */
/*  TX76  << 2  >> TDS :  tds  SUPERVISION DEACTIVATED  */

/*  TX77  << 1  >> TDS :  tds  SUPERVISION CANNOT BE ACTIVATED gr4  */
/*  TX77  << 2  >> TDS :  tds  SUPERVISION ABNORMALLY DEACTIVATED gr4  */

USM /*  TX78  << 1  >> TDS :  tds  ERROR LEVEL  errlvl  DETECTED BY SUPERVISION OF
                       SESSION  */
USM /*  TX78  << 2  >> TDS :  tds  ERROR LEVEL  errlvl  DETECTED BY SUPERVISION OF
                       PROCESS  */

USM /*  TX79  << 01 >> cmdnm  :   NO SPAWNED TRANSACTION ON THE SPECIFIED USER  */

/*  TX80  << 1  >> cmdnm  :  cornm  CONNECTED  */
/*  TX80  << 2  >> cmdnm  :  cornm  NOT CONNECTED; REASON :  reason  */
```

B. Example Application in GPL and C

B.1 Application in GPL

The first part of this appendix contains a listing of an application written in GPL.

Description of the Program

The function performed by this program is to cancel the outputs related to the submitter of the job.

The procedure begins with the declaration of the variables used.

The exchanged structured records which have a meaning for this application are declared by means of the H_DCPMSXR macro-declarative. The DOF 7-PO connection semaphore is declared by means of the H_DCSEM macro-declarative.

The main flow of the program consists of calls to the three internal procedures INITIATION, TREATMENT and TERMINATION. The names are self-explanatory.

The TREATMENT procedure consists of the following:

- Send the command "DJ OUT".
- Wait for the notification of the "responses available" event.
- Get the responses of "DJ OUT" : for each response giving the RON of an output. The CANCEL_OUTPUT procedure is then called.

The CANCEL_OUTPUT procedure consists of the following:

- Send the command "CO <ron>".
- Wait for the notification of the "responses available" event.
- Purge the response of "CO <ron>".

Listing of the Program

```
T_PMOS_EXAMPLE: PROC;

/* ------------------------------------------------------------ */
/*                                                              */
/*                     COMPILE TIME STATEMENTS                  */
/*                                                              */
/* ------------------------------------------------------------ */

%REPLACE R_MSG_SECRI          BY "0001"X;
%REPLACE R_CMD_SECRI          BY "0002"X;
%REPLACE R_LNK_RI             BY "FFF"X;

%REPLACE R_FALSE              BY "0"B;
%REPLACE R_TRUE               BY "1"B;

%REPLACE R_JSTATE_OUT         BY 6;

%REPLACE R_MAXDESCLN          BY 512;
%REPLACE R_MAXRECLN           BY 2048;
%REPLACE R_MAXFMPARLN         BY 256;


/* ------------------------------------------------------------ */
/*                                                              */
/*                            CONSTANTS                         */
/*                                                              */
/* ------------------------------------------------------------ */

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'CDJ_'
                     RECNAME = 'DJ'
                     GEN = DESCRIPTOR
                     TYPE = CMD
                     ;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'CCO_'
                     RECNAME = 'CO'
                     GEN = DESCRIPTOR
                     TYPE = CMD
                     ;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'CSH14_'
                     RECNAME = 'SH14'
                     GEN = DESCRIPTOR
                     TYPE = MSG
                     ;
```

```
/* ---------------------------------------------------------- */
/*                                                            */
/*                          EXTERNAL VARIABLES                */
/*                                                            */
/* ---------------------------------------------------------- */

/* ---------------------------------------------------------- */
/*             DECLARE THE CONNECTION SEMAPHORE               */
/* ---------------------------------------------------------- */

          $H_DCSEM PREFIX = SEM
                   CNT1 = 0
                   MCNT1 = 16
                   TYPE1 = MSGPRTY
                   ;


/* ---------------------------------------------------------- */
/*                                                            */
/*                         LOCAL VARIABLES                    */
/*                                                            */
/* ---------------------------------------------------------- */

DCL L_SEM                    PTR;
DCL L_PMSID                  LOGBIN (32) BYTE;
DCL L_CMDID                  LOGBIN (16) BYTE;
DCL L_MSGID                  LOGBIN (16) BYTE;

DCL L_STATE                  LOGBIN (16) BYTE;
DCL 1 *                      DEFINED L_STATE,
      2 L_STATE_ABN          BIT (1),
      2 *                    BIT (4),
      2 L_STATE_STATUS       BIT (3),
      2 L_STATE_DATA         BIT (8);

DCL L_SAVE_STATE             LOGBIN (16) BYTE;
DCL L_SAVE_RETCODE           LOGBIN (32) BYTE;

DCL L_CURRENT_RECNAME        CHAR (8);
DCL L_CURRENT_DESCPTR        PTR;

DCL L_SEMMSG                 CHAR (16);
DCL L_SAVE_SEMMSG            CHAR (16);

DCL L_DESCLN                 FIXED BIN (15);
DCL L_RECLN                  FIXED BIN (15);
DCL L_FMPARLN                FIXED BIN (15);

DCL L_DESC                   CHAR (R_MAXDESCLN);
DCL L_COMMAND                CHAR (R_MAXRECLN);
DCL L_RECORD                 CHAR (R_MAXRECLN);
DCL L_FMPAR                  CHAR (R_MAXFMPARLN);
```

```
DCL 1 L_FLAGS,
      2 L_INITIATED_CONNECTION BIT (1),
      2 L_RESPONSE_AVAILABLE  BIT (1),
      2 *                     BIT (6);

          $H_DCJOBIDENT PREFIX= JOBIDENT_
                        ;
/* ------------------------------------------------------------ */
/*                                                              */
/*                         BASED VARIABLES                      */
/*                                                              */
/* ------------------------------------------------------------ */

          $H_DCPMSSEMMSG PREFIX = L_CMD_
                         EVENT = CMD
                         ATTRIB = 'BASED(ADDR(L_SEMMSG))'
                         ;

          $H_DCPMSSEMMSG PREFIX = L_MSG_
                         EVENT = MSG
                         ATTRIB = 'BASED(ADDR(L_SEMMSG))'
                         ;

          $H_DCPMSSEMMSG PREFIX = L_LNK_
                         EVENT = LNK
                         ATTRIB = 'BASED(ADDR(L_SEMMSG))'
                         ;

          $H_DCPMSXR ATTRIB = 'BASED(ADDR(L_COMMAND))'
                     PREFIX = 'BDJ_'
                     RECNAME = 'DJ'
                     GEN = RECORD
                     TYPE = CMD
                     ;

          $H_DCPMSXR ATTRIB = 'BASED(ADDR(L_COMMAND))'
                     PREFIX = 'BCO_'
                     RECNAME = 'CO'
                     GEN = RECORD
                     TYPE = CMD
                     ;

          $H_DCPMSXR ATTRIB = 'BASED(ADDR(L_RECORD))'
                     PREFIX = 'BSH14_'
                     RECNAME = 'SH14'
                     GEN = RECORD
                     TYPE = MSG
                     ;

          $H_DCPMSKDESC ATTRIB = 'BASED(ADDR(L_DESC))'
                        PREFIX = 'DESC_'
                        ;
```

```
/* ------------------------------------------------------------ */
/*                                                              */
/*                            MAIN FLOW                         */
/*                                                              */
/* ------------------------------------------------------------ */

          CALL INITIATION;
          IF L_INITIATED_CONNECTION = R_FALSE
          THEN GOTO END_OF_EXAMPLE;

          IF $H_TESTRC NORMAL; THEN CALL TREATMENT;

          CALL TERMINATION;

END_OF_EXAMPLE:
          RETURN;
/* ------------------------------------------------------------ */
/*                                                              */
/*                     INTERNAL PROCEDURES                      */
/*                                                              */
/* ------------------------------------------------------------ */


/* ------------------------------------------------------------ */
/*                                                              */
/*                        CANCEL_OUTPUT                         */
/*                                                              */
/* ------------------------------------------------------------ */

CANCEL_OUTPUT: PROC (PI__RON);

/* RON OF THE OUTPUT TO BE CANCELLED */
DCL PI__RON                   CHAR (8);

/* ------------------------------------------------------------ */
/*               SEND THE COMMAND "CANCEL_OUTPUT"               */
/* ------------------------------------------------------------ */

/* RESET THE FLAG  */
              L_RESPONSE_AVAILABLE = R_FALSE;
/* NEW COMMAND IDENTIFIER */
              L_CMDID = L_CMDID + 1;
/* RESET THE PRESENCE MASK */
              SUBSTR (STRING (BCO_PMOS_CO_REC), 1,
                      MEASURE (BCO_PRESENCE_MASK))
              = REPEAT ("00"H, MEASURE (BCO_PRESENCE_MASK) - 1);
/* VALIDATE THE COMMAND RECORD */
              BCO_RON1_ON = R_TRUE;
              BCO_RON1 = PI__RON;
              BCO_ANY_ON = R_TRUE;
              BCO_ANY = "1";
```

```
/* SEND THE REQUEST TO THE OBJECT MANAGER */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = CCO_PMOS_CO_HD
                             DESCLN = 'MEASURE(CCO_PMOS_CO_HD)'
                             REC = BCO_PMOS_CO_REC
                             RECLN = 'MEASURE(BCO_PMOS_CO_REC)'
                             ;
               IF $H_TESTRC ABNORMAL;
               THEN GOTO END_OF_CANCEL_OUTPUT;

/* ------------------------------------------------------------ */
/*      WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE"       */
/* ------------------------------------------------------------ */

               DO UNTIL (L_RESPONSE_AVAILABLE = R_TRUE);
                    CALL SEPM (ADDR (SEM1), L_CMD_SEMMSG);
                    IF L_CMD_SEMMSG_RI ^= "000"X
                    THEN GOTO END_OF_CANCEL_OUTPUT;
                    SELECT (L_CMD_SEMMSG_SECRI);
                         WHEN (R_MSG_SECRI) CALL GET_MESSAGE;
                         WHEN (R_CMD_SECRI) L_RESPONSE_AVAILABLE
                                            = R_TRUE;
                         OTHERWISE GOTO END_OF_CANCEL_OUTPUT;
                    END;
               END;

/* ------------------------------------------------------------ */
/*            PURGE THE RESPONSES OF "CANCEL_OUTPUT"            */
/* ------------------------------------------------------------ */

               DO UNTIL (($H_TESTRC ABNORMAL;)
                         ! (L_STATE_STATUS = "100"B));
                    $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                                CMDID = L_CMD_SEMMSG_CMDID
                                MAXDESCLN = 'MEASURE(L_DESC)'
                                DESC = L_DESC
                                DESCLN = L_DESCLN
                                MAXRECLN = 'MEASURE(L_RECORD)'
                                REC = L_RECORD
                                RECLN = L_RECLN
                                STATE = L_STATE
                                ;
               END;

END_OF_CANCEL_OUTPUT:
          END CANCEL_OUTPUT;
/* ------------------------------------------------------------ */
/*                                                              */
/*                          GET_MESSAGE                         */
/*                                                              */
/* ------------------------------------------------------------ */

GET_MESSAGE: PROC;
```

```
/* ---------------------------------------------------------- */
/*                      PURGE THE MESSAGES                    */
/* ---------------------------------------------------------- */

              DO UNTIL (($H_TESTRC ABNORMAL;)
                        ! (L_STATE_STATUS = "100"B));
                   $H_PMSGETMSG PMSID = L_MSG_SEMMSG_PMSID
                                FLSID = L_MSG_SEMMSG_FLSID
                                MSGID = L_MSGID
                                MAXDESCLN = 'MEASURE(L_DESC)'
                                DESC = L_DESC
                                DESCLN = L_DESCLN
                                MAXRECLN = 'MEASURE(L_RECORD)'
                                REC = L_RECORD
                                RECLN = L_RECLN
                                STATE = L_STATE
                                ;
              END;

         END GET_MESSAGE;


/* ---------------------------------------------------------- */
/*                                                            */
/*                          INITIATION                        */
/*                                                            */
/* ---------------------------------------------------------- */

INITIATION: PROC;
              STRING (L_FLAGS) = "0"B;

/* ---------------------------------------------------------- */
/*                 INITIATE THE CONNECTION                    */
/* ---------------------------------------------------------- */

              $H_PMSOPEN SEM = 'ADDR (SEM1)'
                         USERNAME = "OPERATOR"
                         PASSWORD = "OP"
                         PMSID = L_PMSID
                         CLEAN
                         ;
              IF $H_TESTRC ABNORMAL;
              THEN GOTO END_OF_INITIATION;

              L_INITIATED_CONNECTION = R_TRUE;

/* ---------------------------------------------------------- */
/*            WAIT FOR THE INITIATION NOTIFICATION            */
/* ---------------------------------------------------------- */

              DO UNTIL (L_LNK_SEMMSG_RI = R_LNK_RI);
                   CALL SEPM (ADDR (SEM1), L_LNK_SEMMSG);
              END;
```

```
/* ---------------------------------------------------------- */
/*          GET INFORMATION ABOUT JOB IDENTIFICATION          */
/* ---------------------------------------------------------- */

              $H_JOBIDENT JOBIDENT_JOBIDENT;

END_OF_INITIATION:
          END INITIATION;


/* ---------------------------------------------------------- */
/*                                                            */
/*                         TERMINATION                        */
/*                                                            */
/* ---------------------------------------------------------- */

TERMINATION: PROC;

/* ---------------------------------------------------------- */
/*                 TERMINATE THE CONNECTION                   */
/* ---------------------------------------------------------- */


              $H_PMSCLOSE PMSID = L_PMSID
                          ;

          END TERMINATION;


/* ---------------------------------------------------------- */
/*                                                            */
/*                           TREATMENT                        */
/*                                                            */
/* ---------------------------------------------------------- */

TREATMENT: PROC;

/* ---------------------------------------------------------- */
/*                SEND THE COMMAND "DISPLAY_JOB"              */
/* ---------------------------------------------------------- */

/* COMMAND IDENTIFIER */
              L_CMDID = "0001"X;
/* RESET THE PRESENCE MASK                                  */
              SUBSTR (STRING (BDJ_PMOS_DJ_REC), 1,
                      MEASURE (BDJ_PRESENCE_MASK))
              = REPEAT ("00"H, MEASURE (BDJ_PRESENCE_MASK) - 1);
/* VALIDATE THE COMMAND RECORD */
              BDJ_JSTATE_ON = R_TRUE;
              BDJ_JSTATE = R_JSTATE_OUT;
              BDJ_USERNM_ON = R_TRUE;
              BDJ_USERNM = JOBIDENT_USER;
```

```
/* SEND THE REQUEST TO THE OBJECT MANAGER */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = CDJ_PMOS_DJ_HD
                             DESCLN = 'MEASURE(CDJ_PMOS_DJ_HD)'
                             REC = BDJ_PMOS_DJ_REC
                             RECLN = 'MEASURE(BDJ_PMOS_DJ_REC)'
                             ;
               IF $H_TESTRC ABNORMAL; THEN GOTO END_OF_TREATMENT;

/* ---------------------------------------------------------- */
/*      WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE"      */
/* ---------------------------------------------------------- */

               DO UNTIL (L_RESPONSE_AVAILABLE = R_TRUE);
                    CALL SEPM (ADDR (SEM1), L_CMD_SEMMSG);
                    SELECT (L_CMD_SEMMSG_SECRI);
                         WHEN (R_MSG_SECRI) CALL GET_MESSAGE;
                         WHEN (R_CMD_SECRI) L_RESPONSE_AVAILABLE
                                            = R_TRUE;
                         OTHERWISE GOTO END_OF_TREATMENT;
                    END;
               END;
/* ---------------------------------------------------------- */
/*              GET THE RESPONSES OF "DISPLAY_JOB"             */
/* ---------------------------------------------------------- */


LOOP:          DO UNTIL (L_STATE_STATUS = "100"B);
                    L_CURRENT_DESCPTR
                    = ADDR (CSH14_PMOS_SH14_HD);
                    $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                                CMDID = L_CMD_SEMMSG_CMDID
                                MAXDESCLN = 'MEASURE(L_DESC)'
                                DESC = L_DESC
                                DESCLN = L_DESCLN
                                MAXRECLN = 'MEASURE(L_RECORD)'
                                REC = L_RECORD
                                RECLN = L_RECLN
                                STATE = L_STATE
                                CURDESCPTR = L_CURRENT_DESCPTR
                                NEXTRECNAME = L_CURRENT_RECNAME
                                ;
                    IF $H_TESTRC ABNORMAL;
                    THEN LEAVE LOOP;
```

```
                        IF (($H_TESTRC DONE;) ! ($H_TESTRC DATALIM;))
                        THEN DO;
                             IF ((DESC_RECNAME = "SH14")
                                 & (BSH14_RON_ON = R_TRUE))
                             THEN DO;
                                  L_SAVE_SEMMSG = L_SEMMSG;
                                  L_SAVE_STATE = L_STATE;
                                  CALL CANCEL_OUTPUT (BSH14_RON);
                                   L_STATE = L_SAVE_STATE;
                                   L_SEMMSG = L_SAVE_SEMMSG;
                             END;
                        END;
/* END OF DO LOOP */
                   END LOOP;

END_OF_TREATMENT:
              END TREATMENT;


END_OF_PROCEDURE:;
     END T_PMOS_EXAMPLE;
```

B.2 Application in C

The aim of this application is to cancel the outputs related to the submitter of the job.

The exchanged structured records which have a meaning for this application are defined by the following header files:

<cmd_CO.h>

<cmd_DJ.h>

<msg_SH14.h>

These header files are described below.

The cmd_CO.h Header File

The <cmd_CO.h> file contains the type declaration and the initialization function of the descriptor associated with the CO DOF 7-PO command. It also contains the type declaration of the record associated with CO. The CO command corresponds to the CANCEL_OUTPUT GCL command. For details, refer to the *GCOS7 Structured Records* Reference Manual.

```
struct _CO_DESC {
     struct _pms_kdesc kdesc;
     char desc1[34];
     char desc2[34];
     char desc3[34];
     char desc4[34];
     char desc5[34];
     char desc6[3];
};

{ h_init_CO_DESC(descptr) }
struct _CO_DESC *descptr;

struct _CO_REC {
     struct {
          struct {
               char p1alter_on;
               char ron1_on;
               char ron2_on;
               char ron3_on;
               char ron4_on;
               char ron5_on;
               char ron6_on;
               char ron7_on;
               char ron8_on;
               char ron9_on;
```

```
                char ron10_on;
                char beg_selclass_on;
                char end_selclass_on;
                char any_on;
                char ounm_on;
                char ouseqnb_on;
                char p2alter_on;
                char strong_on;
            } present;
            char p1alter;
            char ron1[8];
            char ron2[8];
            char ron3[8];
            char ron4[8];
            char ron5[8];
            char ron6[8];
            char ron7[8];
            char ron8[8];
            char ron9[8];
            char ron10[8];
            char beg_selclass[3];
            char end_selclass[3];
            char any;
            char ounm[8];
            short ouseqnb;
            char p2alter;
            char strong;
        } sel;
};
```

The cmd_DJ.h Header file

The <cmd_DJ.h> file contains the type declaration and the initialization function of the descriptor associated with the DJ DOF 7-PO command. It also contains the type declaration of the record associated with DJ. The DJ command corresponds to the DISPLAY_JOB GCL command. For details, refer to the *GCOS7 Structured Records Reference Manual.*

```
struct _DJ_DESC {
     struct _pms_kdesc kdesc;
     char desc1[34];
     char desc2[34];
     char desc3[26];
};

{ h_init_DJ_DESC(descptr) }
struct _DJ_DESC *descptr;
```

```
struct _DJ_REC {
     struct {
          struct {
               char p1alter_on;
               char ron_on;
               char jnm_on;
               char jstate_on;
               char jobtype_on;
               char projnm_on;
               char usernm_on;
               char submitter_on;
               char jclass_on;
               char limit_on;
          } present;
          char p1alter;
          char ron[8];
          char jnm[8];
          char jstate;
          char jobtype[8];
          char projnm[12];
          char usernm[12];
          char submitter[12];
          char jclass[2];
          long limit;
     } sel;
};
```

The msg_SH14.h Header file

The <msg_SH14.h> file contains the type declaration and the initialization function of the descriptor associated with the SH14 message. It also contains the type declaration of the record associated with the SH14 message. The SH14 message may be received in response to the DJ DOF 7-PO command. For details, refer to *Structured Records (OMH Format) Part 2 - Messages.*

```
struct _SH14_DESC {
     struct _pms_kdesc kdesc;
     char desc1[34];
     char desc2[34];
     char desc3[20];
};

{ h_init_SH14_DESC(descptr) }
struct _SH14_DESC *descptr;
```

```
struct _SH14_REC {
     struct {
          struct {
               unsigned p1alter_on : 1;
               unsigned ron_on : 1;
               unsigned jstate_on : 1;
               unsigned jnm_on : 1;
               unsigned usernm_on : 1;
               unsigned jclass_on : 1;
               unsigned ssn_on : 1;
               unsigned dimname_on : 1;
               unsigned lm_name_on : 1;
               unsigned filler : 7;
          } present;
          char p1alter;
          char ron[8];
          char jstate;
          char jnm[8];
          char usernm[12];
          char jclass[2];
          short ssn;
          char dimname[8];
          char lm_name[32];
     } resp;
};
```

Description of the Program

The DOF 7-PO connection semaphore is obtained from a semaphore pool created by the application (see the *C Primitives* Manual for details).

The main flow of this program consists of the call of five functions: initiation, get_jobident, display_job, get_response, termination. The names are self-explanatory.

The get_jobident function gets the identification name of the job submitter.

The display_job function sends the command "DJ JOB_STATE=OUT".

The get_response function gets the responses of "DJ OUT". For each response giving the Run Occurrence Number (RON) of an output, the cancel_output function is then called.

The cancel_output function consists of the following:

- send the command "CO ron";
- wait for the notification of the "responses available event";
- purge and edit the responses of "CO ron".

Listing of the Program

```
#include <retcode.h>
#include <jobm.h>
#include <pmos.h>
#include <cmd_CO.h>
#include <cmd_DJ.h>
#include <msg_SH14.h>
#include <taskm.h>
#define MSG_RI      1      /* unsolicited message request id. */
#define CMD_RI      2      /* response request identifier     */
#define LNK_PRTY    15     /* event enqueueing priority       */
#define MSG_PRTY    15     /* message enqueueing priority     */
#define CMD_PRTY    15     /* response enqueueing priority    */
#define TRUE        1
#define FALSE       0
#define JSTATE_OUT  6
#define MAXDESCLN   512
#define MAXRECLN    2048
#define MSGLN       78

unsigned long pmsid, cmdid=1, cmdid_DJ;
char desc[MAXDESCLN];
char rec[MAXRECLN];
char *sem, *pool, event_semmsg[16];
char *station=PMS_LOCAL_STATION;
struct _jobident jobident;


/*
   This program cancels the outputs related
   to the submitter of the job
*/
main()
{
   struct _pms_cmdsemmsg *cmdsemmsg;

   initiation();

   get_jobident();

   display_job();
```

```
   /* wait for the notification of "response available"    */
   cmdsemmsg = (struct _pms_cmdsemmsg *) event_semmsg;
   do {
      h_sepm(sem,event_semmsg);
      switch (cmdsemmsg->cmdri) {
         case CMD_RI : break;
         case MSG_RI : get_message();
                       break;
         default:      exit(1);
      }
   } while (cmdsemmsg->cmdri != CMD_RI);

   get_response();
   termination();
}
/*
  Create a semaphore for DOF 7-PO events notifications
  and initiate a connection
*/
initiation()
{
   struct _pms_sem pms_sem;
   struct _pms_lnksemmsg *lnksemmsg;

   h_crsempool(pool,1,0,128);
   if (!h_testrc(DONE)) stop("Unable to create semaphore pool");

   /* get a semaphore with message from pool,              */
   /* initial count=0, maximum count=16                    */
   { h_getsem (sem,pool,0,16,SM); }
   if (!h_testrc(DONE)) stop("Unable to get semaphore");

   /* initiate the connection                          */
   pms_sem.sem=sem;
   pms_sem.msgri=MSG_RI;
   pms_sem.msgprty=MSG_PRTY;
   pms_sem.cmdri=CMD_RI;
   pms_sem.cmdprty=CMD_PRTY;
   pms_sem.lnkprty=LNK_PRTY;
   { h_pmsopen(pms_sem,0,0,PMS_LOCAL_SITE,pmsid); }
   if (!h_testrc(DONE)) stop("Unable to open DOF 7-PO");

   /* wait for the initiation notification                */
   lnksemmsg = (struct _pms_lnksemmsg *) event_semmsg;
   do
      h_sepm(sem,event_semmsg);
   while ( (lnksemmsg->sysri != PMS_LNKRI )
         || (lnksemmsg->pmsid != pmsid));
}
```

```
/* get information about job identification                */
get_jobident()
{
   { h_jobident(jobident,SHORT); }
}
/* send the command "DISPLAY_JOB JOB_STATE=OUT"            */
display_job()
{
   int i;
   struct _pms_param pms_param;
   struct _DJ_DESC *DJ_descptr;
   struct _DJ_REC *DJ_recptr;
   DJ_descptr= (struct _DJ_DESC *) desc;
   DJ_recptr = (struct _DJ_REC *) rec;

   /* initialize the command descriptor                    */
   { h_init_DJ_DESC (DJ_descptr); }
   /* reset the presence mask to zeroes                    */
   for (i=0; i<sizeof(DJ_recptr->sel.present); i++)
      *(((char *) DJ_recptr) + i) = '\0';

   /* validate the command record                          */
   DJ_recptr->sel.present.jstate_on=TRUE;
   DJ_recptr->sel.jstate=JSTATE_OUT;
   DJ_recptr->sel.present.usernm_on=TRUE;
   cpybuf(DJ_recptr->sel.usernm,jobident.user,12);

   /* send the request to the object manager               */
   cmdid_DJ = cmdid++;
   pms_param = _pms_param_init;
   pms_param.descptr = (char *) DJ_descptr;
   pms_param.descln = sizeof(*DJ_descptr);
   pms_param.recptr = (char *) DJ_recptr;
   pms_param.recln = sizeof(*DJ_recptr);
   edit_message(pms_param);
   { h_pmssendcmd(pmsid,cmdid_DJ,pms_param,station,
   -1,PMS_DEFAULT_SITE); }
   if (!h_testrc(DONE)) stop("Unable to send command DJ");
}


/* edit a command or a response given in a structured record */
edit_message(pms_param)
struct _pms_param pms_param;
{
   struct { short length; char text[MSGLN+1]; } message;
   { h_pmsedtmsg(pms_param,message,MSGLN,MSG); }
   printf("%s\n",message.text);
}
```

```
/*
  Get the expected responses SH14 of DISPLAY_JOB
  and cancel each returned output
*/
get_response()
{
   struct _SH14_DESC *SH14_descptr, SH14_desc;
   struct _SH14_REC *SH14_recptr;
   struct _pms_param pms_param;
   struct _pms_qstatus state;
   char nextrecname[8], ron[8];

   SH14_descptr= (struct _SH14_DESC *) desc;
   SH14_recptr = (struct _SH14_REC *) rec;
   { h_init_SH14_DESC(&SH14_desc); }
   do {
      pms_param = _pms_param_init;
      pms_param.maxdescln = MAXDESCLN;
      pms_param.descptr = (char *) desc;
      pms_param.maxrecln =  MAXRECLN;
      pms_param.recptr = (char *) rec;
      pms_param.curdescptr = (char *) &SH14_desc;
      { h_pmsgetrp (pmsid,cmdid_DJ,pms_param,nextrecname,state); }
      if (h_testrc(DONE) || h_testrc(DATALIM)) {
         if (SH14_recptr->resp.present.ron_on) {
            pms_param.curdescptr = NULL_PTR;
            pms_param.curdescln = 0;
            pms_param.maxdescln = 0;
            pms_param.maxrecln = 0;
            edit_message(pms_param);
            cpybuf(ron,SH14_recptr->resp.ron,8);
            cancel_output((char *)ron);
         }
      }
      else {
         if (h_testrc(ABNORMAL)) stop("Unable to get DJ
         response");
      }
  } while ( (!state.abnormal) && (state.delivered != PMS_EMPTY) );
}
/* purge the unsolicited messages */
get_message()
{
   struct _pms_msgsemmsg *msgsemmsg;
   struct _pms_param pms_param;
   struct _pms_qstatus state;
   unsigned short flsid, msgid;
   char nextrecname[8];
```

```
   msgsemmsg = (struct _pms_msgsemmsg *) event_semmsg;
   flsid = msgsemmsg->flsid;
   do {
      pms_param = _pms_param_init;
      pms_param.maxdescln = MAXDESCLN;
      pms_param.descptr = (char *) desc;
      pms_param.maxrecln =  MAXRECLN;
      pms_param.recptr = (char *) rec;
     {h_pmsgetmsg(pmsid,flsid,msgid,pms_param,nextrecname,state);}
  } while ( (!state.abnormal) && (state.delivered != PMS_EMPTY) );
}


/* send the command "CANCEL_OUTPUT" of the specified RON   */
cancel_output(ron)
char *ron;
{
   int i;
   unsigned long cmdid_CO;
   char nextrecname[8];
   struct _pms_cmdsemmsg *cmdsemmsg;
   struct _pms_param pms_param;
   struct _pms_qstatus state;
   struct _CO_DESC *CO_descptr;
   struct _CO_REC *CO_recptr;
   CO_descptr= (struct _CO_DESC *) desc;
   CO_recptr = (struct _CO_REC *) rec;

   /* initialize the command descriptor                    */
   { h_init_CO_DESC (CO_descptr); }

   /* reset the presence mask to zeroes                    */
   for (i=0; i<sizeof(CO_recptr->sel.present); i++)
      *(((char *) CO_recptr) + i) = '\0';

   /* validate the command record                          */
   CO_recptr->sel.present.ron1_on=TRUE;
   cpybuf(CO_recptr->sel.ron1,ron,8);
   CO_recptr->sel.present.any_on=TRUE;
   CO_recptr->sel.any='1';
   /* send the request to the object manager               */
   cmdid_CO = cmdid++;
   pms_param = _pms_param_init;
   pms_param.descptr = (char *) CO_descptr;
   pms_param.descln = sizeof(*CO_descptr);
   pms_param.recptr = (char *) CO_recptr;
   pms_param.recln = sizeof(*CO_recptr);
   edit_message(pms_param);
   { h_pmssendcmd(pmsid,cmdid_CO,pms_param,station,
    -1,PMS_DEFAULT_SITE); }
   if (!h_testrc(DONE)) stop("Unable to send command CO");
```

```
   /* wait for the notification of "response available"    */
   cmdsemmsg = (struct _pms_cmdsemmsg *) event_semmsg;
   do {
      h_sepm(sem,event_semmsg);
      switch (cmdsemmsg->cmdri) {
         case CMD_RI : break;
         case MSG_RI : get_message();
                       break;
         default:      exit(1);
      }
   } while (cmdsemmsg->cmdri != CMD_RI);

   /* purge the responses of CANCEL_OUTPUT                */
   do {
      pms_param = _pms_param_init;
      pms_param.maxdescln = MAXDESCLN;
      pms_param.descptr = (char *) desc;
      pms_param.maxrecln =  MAXRECLN;
      pms_param.recptr = (char *) rec;
      { h_pmsgetrp (pmsid,cmdid_CO,pms_param,nextrecname,state); }
      pms_param.curdescptr = NULL_PTR;
      pms_param.curdescln = 0;
      pms_param.maxdescln = 0;
      pms_param.maxrecln = 0;
      edit_message(pms_param);
  } while ( (!state.abnormal) && (state.delivered != PMS_EMPTY) );
}


/* terminate the connection and free the semaphore    */
termination()
{
   { h_pmsclose (pmsid,PMS_LOCAL_SITE); }
   if (!h_testrc(DONE)) stop("Unable to close connection");

   { h_freesem(sem,pool,SM); }
   if (!h_testrc(DONE)) stop("Unable to free semaphore");
   h_dlsempool(pool);

   if (!h_testrc(DONE)) stop("Unable to delete semaphore pool");
}
/* exit routine called when an abnormal return code is returned */
stop(message)
char *message;
{
   char rc[31];
   { h_editrc(rc); }
   printf("%s, %s\n",message,rc);
   exit(1);
}
```

C. Example Application Using Filters

The aim of this example is to show the mechanism of filters and the reception of unsolicited messages.

A main operator:

- Opens a DOF 7-PO session (only MAIN users can manage filters and receive generic messages).
- Creates a filter set and an inclusive filter in it. The filter will enable the reception of JB08 messages related to the execution of the job, which is called PMOS_JOB.
- Starts the job to be executed. The job PMOS_JOB is stored in the PMOS.SLLIB source library. It consists of four steps (so four JB08s are expected).
- Loops to receive unsolicited messages. If the notification received is for:
  - specific messages (FLSID = "4040"X), the chain of messages is edited. If the key of the last message received is:

    IN31: NO JOB INTRODUCED, the program is stopped,
    JB02 or OU14: the execution of the job has finished and the program is stopped,
    others: the program waits for the next notification.

  - generic messages (JB08 messages), the chain of messages is edited.
    A counter is incremented. When it is equal to four (the last step is executing), the program is stopped.
- Closes the DOF 7-PO session.

```
EX_FILT_PMOS: PROC;

/* -----------COMPILE TIME STATEMENTS--------------- */

%REPLACE R_MSG_SECRI          BY "0001"X;
%REPLACE R_CMD_SECRI          BY "0002"X;
%REPLACE R_LNK_RI             BY "FFF"X;

%REPLACE R_FALSE              BY "0"B;
%REPLACE R_TRUE               BY "1"B;

%REPLACE R_MAXDESCLN          BY 512;
%REPLACE R_MAXRECLN           BY 2048;
%REPLACE R_MAXFMPARLN         BY 256;


/* ------------- CONSTANTS ----------------------- */

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'FLTST_'
                     RECNAME = 'CRFLTST'
                     GEN = DESCRIPTOR
                     TYPE = CMD;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'FLT_'
                     RECNAME = 'CRFLT'
                     GEN = DESCRIPTOR
                     TYPE = CMD;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'EJR_'
                     RECNAME = 'EJR'
                     GEN = DESCRIPTOR
                     TYPE = CMD;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'FT02_'
                     RECNAME = 'FT02'
                     GEN = DESCRIPTOR
                     TYPE = MSG;

/* ---------------- EXTERNAL VARIABLES ------------- */

          $H_DCSEM PREFIX = SEM
                   CNT1 = 0
                   MCNT1 = 16
                   TYPE1 = MSGPRTY;
```

```
/* --------------- LOCAL VARIABLES ---------------- */

DCL L_SEM                     PTR;
DCL L_PMSID                   LOGBIN (32) BYTE;
DCL L_CMDID                   LOGBIN (16) BYTE;
DCL L_MSGID                   LOGBIN (16) BYTE;

DCL L_STATE                   LOGBIN (16) BYTE;
DCL 1 *                       DEFINED L_STATE,
      2 L_STATE_ABN           BIT (1),
      2 *                     BIT (4),
      2 L_STATE_STATUS        BIT (3),
      2 L_STATE_DATA          BIT (8);

DCL L_CURRENT_RECNAME         CHAR (8);
DCL L_CURRENT_DESCPTR         PTR;

DCL L_SEMMSG                  CHAR (16);

DCL L_DESCLN                  FIXED BIN (15);
DCL L_RECLN                   FIXED BIN (15);

DCL L_DESC                    CHAR (R_MAXDESCLN);
DCL L_COMMAND                 CHAR (R_MAXRECLN);
DCL L_RECORD                  CHAR (R_MAXRECLN);

DCL L_PTR_CMD                 PTR;
DCL L_PTR_REC                 PTR;
DCL L_PTR_SEM                 PTR;

DCL ERROR_MESSAGE             CHAR (80) INIT ("OPENING OF SESSION
                              IMPOSSIBLE");
DCL LNG_80                    FB15 INIT (80);

/* ----------------- BASED VARIABLES ------------- */

          $H_DCPMSSEMMSG PREFIX = L_CMD_
                         EVENT = CMD
                         ATTRIB = 'BASED(L_PTR_SEM)';

          $H_DCPMSSEMMSG PREFIX = L_MSG_
                         EVENT = MSG
                         ATTRIB = 'BASED(L_PTR_SEM)';

          $H_DCPMSSEMMSG PREFIX = L_LNK_
                         EVENT = LNK
                         ATTRIB = 'BASED(L_PTR_SEM)';

          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_CMD)'
                     PREFIX = 'FLTST_'
                     RECNAME = 'CRFLTST'
                     GEN = RECORD
                     TYPE = CMD;
```

```
          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_CMD)'
                     PREFIX = 'FLT_'
                     RECNAME = 'CRFLT'
                     GEN = RECORD
                     TYPE = CMD;

          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_CMD)'
                     PREFIX = 'EJR_'
                     RECNAME = 'EJR'
                     GEN = RECORD
                     TYPE = CMD;

          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_REC)'
                     PREFIX = 'FT02_'
                     RECNAME = 'FT02'
                     GEN = RECORD
                     TYPE = MSG;

          $H_DCPMSKDESC ATTRIB = 'BASED(ADDR(L_DESC))'
                        PREFIX = 'DESC_';

/* ------------------ MAIN FLOW --------------- */

          L_PTR_SEM = ADDR (L_SEMMSG);
          L_PTR_CMD = ADDR (L_COMMAND);
          L_PTR_REC = ADDR (L_RECORD);

          IF INITIATION () THEN

          IF CREATE_FILTER () THEN DO;

               IF FOLLOW_THE_JOB () THEN;
               CALL TERMINATION;
          END;

          ELSE CALL TERMINATION;
          ELSE
          $H_PUTIOF MESSAGE = 'ADDR(ERROR_MESSAGE)'
                    LENGTH = LNG_80
                    ;

END_OF_EXAMPLE:
          RETURN;
```

```
/* ---------- INITIATION ------------ */

INITIATION: PROC RETURNS (BIT (1));

/* --------- INITIATE THE CONNECTION --------------- */

               $H_PMSOPEN SEM = ADDR(SEM1)
                          PMSID = L_PMSID
                          USERNAME = "OPERATOR"
                          PASSWORD = "OP"
                          CLEAN
                          ;

               IF $H_TESTRC ABNORMAL;
               THEN RETURN (R_FALSE);

/* --------- WAIT FOR THE INITIATION NOTIFICATION --- */

               DO UNTIL (L_LNK_SEMMSG_RI = R_LNK_RI);
                    CALL SEPM (ADDR (SEM1), L_SEMMSG);
               END;

               RETURN (R_TRUE);

          END;

/* ----- CREATE_FILTER -------- */

CREATE_FILTER: PROC RETURNS (BIT (1));

DCL BOOL_CRFLTST              BIT (1);
DCL RESP_CRFLT                LOGBIN (16);
DCL 1 *                       DEFINED RESP_CRFLT,
      2 NB_P1AL               LOGBIN (8),
      2 RC                    BIT (1),
      2 *                     BIT (7);
/* --- SEND THE COMMAND "CREATE FILTERSET " ----- */

/* -- COMMAND IDENTIFIER --- */
               L_CMDID = "0001"X;

/* --- RESET THE PRESENCE MASK ---- */
              SUBSTR (STRING (FLTST_PMOS_CRFLTST_REC), 1,
              MEASURE (FLTST_PRESENCE_MASK))
              = REPEAT ("00"H, MEASURE (FLTST_PRESENCE_MASK) - 1);

/* VALIDATE THE COMMAND RECORD -- */
               FLTST_FLTST_ON = R_TRUE;
               FLTST_FLTST = "AA";
               FLTST_MSGPRTY_ON = R_TRUE;
               FLTST_MSGPRTY = 0;
```

```
/* --- SEND THE REQUEST TO THE OBJECT MANAGER -- */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = FLTST_PMOS_CRFLTST_HD
                             DESCLN =
                             'MEASURE(FLTST_PMOS_CRFLTST_HD)'
                             REC = FLTST_PMOS_CRFLTST_REC
                             RECLN =
                             'MEASURE(FLTST_PMOS_CRFLTST_REC)'
                             ;

               IF $H_TESTRC ABNORMAL;
               THEN RETURN (R_FALSE);

/* --- WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE -- */

               DO UNTIL (L_CMD_SEMMSG_SECRI = R_CMD_SECRI);
                    CALL SEPM (ADDR (SEM1), L_SEMMSG);
               END;

               RESP_CRFLT = TEST_RESP ();
               IF ((RC ^= R_TRUE) ! (NB_P1AL ^= 4))
               THEN RETURN (R_FALSE);

               IF ^EDIT_MSG () THEN RETURN (R_FALSE);

/* -- COMMAND IDENTIFIER --- */
               L_CMDID = L_CMDID + 1;

/* --- RESET THE PRESENCE MASK ---- */
               SUBSTR (STRING (FLT_PMOS_CRFLT_REC), 1,
               MEASURE (FLT_PRESENCE_MASK))
               = REPEAT ("00"H, MEASURE (FLT_PRESENCE_MASK) - 1);
/* VALIDATE THE COMMAND RECORD -- */
               FLT_FLTST_ON = R_TRUE;
               FLT_FLTST = "AA";
               FLT_FLT_ON = R_TRUE;
               FLT_FLT = "AA1";
               FLT_FLTTYP_ON = R_TRUE;
               FLT_FLTTYP = 10;
               FLT_MSGTYP_ON = R_TRUE;
               FLT_MSGTYP = 50;
               FLT_MSGKEY1_ON = R_TRUE;
               FLT_MSGKEY1 = "JB08";
               FLT_JNM_ON = R_TRUE;
               FLT_JNM = "PMOS_JOB";

/* --- SEND THE REQUEST TO THE OBJECT MANAGER -- */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = FLT_PMOS_CRFLT_HD
                             DESCLN = 'MEASURE(FLT_PMOS_CRFLT_HD)'
                             REC = FLT_PMOS_CRFLT_REC
                             RECLN = 'MEASURE(FLT_PMOS_CRFLT_REC)'
                             ;
```

```
                IF $H_TESTRC ABNORMAL;
                THEN RETURN (R_FALSE);

/* --- WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE -- */

                DO UNTIL (L_CMD_SEMMSG_SECRI = R_CMD_SECRI);
                     CALL SEPM (ADDR (SEM1), L_SEMMSG);
                END;

                RESP_CRFLT = TEST_RESP ();
                IF ((RC ^= R_TRUE) ! (NB_P1AL ^= 6))
                THEN RETURN (R_FALSE);

                IF ^EDIT_MSG () THEN RETURN (R_FALSE);

                RETURN (R_TRUE);

END_OF_CREATE_FILTER:
          END CREATE_FILTER;

/* ----- GET THE RESPONSE OF CRFLTST AND CRFLT  -------- */

TEST_RESP: PROC RETURNS (LOGBIN (16));

DCL RESP                      LOGBIN (16);
DCL 1 *                       DEFINED RESP,
      2 NB_P1ALTER            LOGBIN (8),
      2 RC_OK                 BIT (1),
      2 *                     BIT (7);

/* -- GET FIRST RESPONSE -- */

                $H_PMSGETRPFR PMSID = L_CMD_SEMMSG_PMSID
                              CMDID = L_CMD_SEMMSG_CMDID
                              RECNAME = L_CURRENT_RECNAME
                              ;

                IF (L_CURRENT_RECNAME = "FT02") THEN DO;

                     L_CURRENT_DESCPTR = ADDR (FT02_PMOS_FT02_HD);

/* -- GET THE RESPONSE OF "CREATE FILTERSET" -- */

                     $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                                 CMDID = L_CMD_SEMMSG_CMDID
                                 MAXDESCLN = 'MEASURE(L_DESC)'
                                 DESC = L_DESC
                                 DESCLN = L_DESCLN
                                 MAXRECLN = 'MEASURE(L_RECORD)'
                                 REC = L_RECORD
                                 RECLN = L_RECLN
                                 STATE = L_STATE
                                 CURDESCPTR = L_CURRENT_DESCPTR
                                 NEXTRECNAME = L_CURRENT_RECNAME
                                 ;
```

```
                    IF (($H_TESTRC DONE;) ! ($H_TESTRC DATALIM;))
                    THEN DO;
                         IF (FT02_P1ALTER_ON = R_TRUE) THEN DO;
                              NB_P1ALTER = FT02_P1ALTER;
                              RC_OK = R_TRUE;
                         END;
                         ELSE RC_OK = R_FALSE;
                    END;
                    ELSE RC_OK = R_FALSE;

               END;
               ELSE

LOOP:
               DO UNTIL (L_STATE_STATUS = "100"B);

                    $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                                CMDID = L_CMD_SEMMSG_CMDID
                                MAXDESCLN = 'MEASURE(L_DESC)'
                                DESC = L_DESC
                                DESCLN = L_DESCLN
                                MAXRECLN = 'MEASURE(L_RECORD)'
                                REC = L_RECORD
                                RECLN = L_RECLN
                                STATE = L_STATE
                                ;

                    IF $H_TESTRC ABNORMAL;
                    THEN RC_OK = R_FALSE;

                    IF ^EDIT_MSG () THEN;
                    RC_OK = R_FALSE;
/* END OF DO LOOP */
               END LOOP;

               RETURN (RESP);

END_OF_TEST_RESP:
          END TEST_RESP;

/* ----- FOLLOW_THE_JOB -------- */

FOLLOW_THE_JOB: PROC RETURNS (BIT (1));

DCL CPT_JB08                  LOGBIN (16) INIT (0);
DCL MSG                       CHAR (9);
DCL 1 *                       DEFINED MSG,
      2 LAST_KEY              CHAR (8),
      2 RC                    BIT (1),
      2 *                     BIT (7);
```

```
/* -- COMMAND IDENTIFIER --- */
               L_CMDID = L_CMDID + 1;

/* --- RESET THE PRESENCE MASK ---- */
               SUBSTR (STRING (EJR_PMOS_EJR_REC), 1,
               MEASURE (EJR_PRESENCE_MASK))
               = REPEAT ("00"H, MEASURE (EJR_PRESENCE_MASK) - 1);

/* VALIDATE THE COMMAND RECORD -- */
               EJR_SFN_ON = R_TRUE;
               EJR_SFN = "PMOS_JOB";
               EJR_EFN_ON = R_TRUE;
               EJR_EFN = "PMOS.SLLIB";

/* --- SEND THE REQUEST TO THE OBJECT MANAGER -- */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = EJR_PMOS_EJR_HD
                             DESCLN = 'MEASURE(EJR_PMOS_EJR_HD)'
                             REC = EJR_PMOS_EJR_REC
                             RECLN = 'MEASURE(EJR_PMOS_EJR_REC)'
                             ;
               IF $H_TESTRC ABNORMAL;
               THEN RETURN (R_FALSE);

/* --- WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE -- */

               DO UNTIL (L_CMD_SEMMSG_SECRI = R_CMD_SECRI);
                    CALL SEPM (ADDR (SEM1), L_SEMMSG);
               END;

/* -- GET FIRST RESPONSE -- */
               $H_PMSGETRPFR PMSID = L_CMD_SEMMSG_PMSID
                             CMDID = L_CMD_SEMMSG_CMDID
                             RECNAME = L_CURRENT_RECNAME
                             ;
               $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                           CMDID = L_CMD_SEMMSG_CMDID
                           MAXDESCLN = 'MEASURE(L_DESC)'
                           DESC = L_DESC
                           DESCLN = L_DESCLN
                           MAXRECLN = 'MEASURE(L_RECORD)'
                           REC = L_RECORD
                           RECLN = L_RECLN
                           STATE = L_STATE
                           ;
               IF (L_CURRENT_RECNAME = "OP66") THEN;
               ELSE DO;
                    IF ^EDIT_MSG () THEN;
                    RETURN (R_FALSE);
               END;
```

```
                DO UNTIL ((CPT_JB08 = 4) ! (LAST_KEY = "JB02")
                      ! (LAST_KEY = "OU14"));

                     CALL SEPM (ADDR (SEM1), L_SEMMSG);
                     IF L_CMD_SEMMSG_RI ^= "000"X
                     THEN RETURN (R_FALSE);
                     IF (L_MSG_SEMMSG_SECRI = R_MSG_SECRI) THEN

                     IF (L_MSG_SEMMSG_FLSID = "4040"X) THEN DO;

/* -------------- NOTIFICATION OF SPECIFIC MESSAGE ------------ */
                      MSG = GET_MESSAGE ();
                      IF (RC = R_FALSE) THEN RETURN (RC);
                      IF (LAST_KEY = "IN31") THEN RETURN (R_FALSE);
                     END;

                     ELSE DO;

/* -------------- NOTIFICATION OF GENERIC MESSAGE : JB08 ------ */
                          CPT_JB08 = CPT_JB08 + 1;
                          MSG = GET_MESSAGE ();
                          IF (RC = R_FALSE) THEN RETURN (RC);
                     END;
                END;

                IF (CPT_JB08 = 4) THEN RETURN (R_TRUE);
                ELSE RETURN (R_FALSE);

END_OF_FOLLOW_JOB:
          END FOLLOW_JOB;

/* ----- GET ALL THE MESSAGES  -------- */

GET_MESSAGE: PROC RETURNS (CHAR (9));

DCL MSG                       CHAR (9);
DCL 1 *                       DEFINED MSG,
      2 KEY                   CHAR (8),
      2 RC                    BIT (1),
      2 *                     BIT (7);


/* -- GET THE CHAIN OF MESSAGES   -- */

LOOP:
                DO UNTIL (L_STATE_STATUS = "100"B);

                     $H_PMSGETMSG PMSID = L_MSG_SEMMSG_PMSID
                                  FLSID = L_MSG_SEMMSG_FLSID
                                  MSGID = L_MSGID
                                  MAXDESCLN = 'MEASURE(L_DESC)'
                                  DESC = L_DESC
                                  DESCLN = L_DESCLN
```

```
                                        MAXRECLN = 'MEASURE(L_RECORD)'
                                        REC = L_RECORD
                                        RECLN = L_RECLN
                                        STATE = L_STATE
                                        ;
                              IF ($H_TESTRC ABNORMAL;) THEN DO;
                                   RC = R_FALSE;
                                   RETURN (MSG);
                              END;
                              ELSE KEY = DESC_RECNAME;

                              IF ^EDIT_MSG () THEN DO;
                                   RC = R_FALSE;
                                   RETURN (MSG);
                              END;
                         END;

                         RC = R_TRUE;
                         RETURN (MSG);

END_OF_GET_MESSAGE:
          END GET_MESSAGE;

/* ----- EDIT THE MESSAGE RECEIVED  -------- */

EDIT_MSG: PROC RETURNS (BIT (1));

DCL L_MAX                     FB15 INIT (1024);
DCL 1 L_MESSAGE,
      2 OUTPUT_LENGTH         FB15,
      2 OUTPUT_MESSAGE        CHAR (1024);

                $H_PMSEDTMSG DESC = L_DESC
                             DESCLN = L_DESCLN
                             REC = L_RECORD
                             RECLN = L_RECLN
                             MESSAGE = L_MESSAGE
                             MAXLN = L_MAX
                             ;

                IF $H_TESTRC ABNORMAL;
                THEN RETURN (R_FALSE);

                $H_PUTIOF MESSAGE = 'ADDR(OUTPUT_MESSAGE)'
                          LENGTH = OUTPUT_LENGTH
                          ;

                IF $H_TESTRC ABNORMAL;
                THEN RETURN (R_FALSE);

                RETURN (R_TRUE);

END_OF_EDIT_MSG:
          END EDIT_MSG;
```

```
/* ------------ TERMINATION ----------------- */

TERMINATION: PROC;

               $H_PMSCLOSE PMSID = L_PMSID;

END_OF_TERMINATION:
          END TERMINATION;

END_OF_EX:
     END;
```

D. Example of Remote Application

The aim of this example is to show the remote functions of DOF 7-PO in the GCOS 7 Version 6 offer.

The application:

- Opens a DOF 7-PO session on a remote site (in this example the site can be Bull only or mixed equipment). The RAEH server must be started on both sites: local and remote.
- The session is opened on the remote site if the return code received with the LNKRI event is equal to zero.
- A DSAC network command is then sent (NBSS - Number of Sessions). The command will execute on the remote site specified by the PMSOPEN primitive.
- While waiting on the semaphore of the connection for the "RESPONSE AVAILABLE" notification, if the event "DOF 7-PO FACILITIES (UN)AVAILABLE" event comes (indicating a network failure, or termination of RAEH on one of the sites), the H_PMSCVSITE primitive is called to get the name of the remote site, and the session is closed.
- Gets the response, and display it on the screen.
- Closes the session.

```
REMOTE_PMOS: PROC;

/* -----------COMPILE TIME STATEMENTS--------------- */

%REPLACE R_MSG_SECRI          BY "0001"X;
%REPLACE R_CMD_SECRI          BY "0002"X;
%REPLACE R_LNK_RI             BY "FFF"X;

%REPLACE R_FALSE              BY "0"B;
%REPLACE R_TRUE               BY "1"B;

%REPLACE R_MAXDESCLN          BY 512;
%REPLACE R_MAXRECLN           BY 2048;
%REPLACE R_MAXFMPARLN         BY 256;

/* ---------------- EXTERNAL VARIABLES ------------- */

          $H_DCSEM PREFIX = SEM
                   CNT1 = 0
                   MCNT1 = 16
                   TYPE1 = MSGPRTY;

/* ------------- CONSTANTS ----------------------- */

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = ''
                     RECNAME = 'NBSS'
                     GEN = DESCRIPTOR
                     TYPE = CMD
                     OBJ = NET;

          $H_DCPMSXR ATTRIB = 'CONSTANT'
                     PREFIX = 'R_'
                     RECNAME = 'NBSS'
                     GEN = DESCRIPTOR
                     TYPE = RSP
                     OBJ = NET;

/* --------------- LOCAL VARIABLES ---------------- */

DCL L_SEM                     PTR;
DCL L_PMSID                   LOGBIN (32) BYTE;
DCL L_CMDID                   LOGBIN (16) BYTE;
DCL L_MSGID                   LOGBIN (16) BYTE;

DCL L_STATE                   LOGBIN (16) BYTE;
DCL 1 *                       DEFINED L_STATE,
      2 L_STATE_ABN           BIT (1),
      2 *                     BIT (4),
      2 L_STATE_STATUS        BIT (3),
      2 L_STATE_DATA          BIT (8);

DCL L_CURRENT_RECNAME         CHAR (8);
DCL L_CURRENT_DESCPTR         PTR;
```

```
DCL L_SEMMSG                    CHAR (16);

DCL L_DESCLN                    FIXED BIN (15);
DCL L_RECLN                     FIXED BIN (15);

DCL L_DESC                      CHAR (R_MAXDESCLN);
DCL L_COMMAND                   CHAR (R_MAXRECLN);
DCL L_RECORD                    CHAR (R_MAXRECLN);

DCL L_PTR_CMD                   PTR;
DCL L_PTR_REC                   PTR;
DCL L_PTR_SEM                   PTR;

DCL ERREUR_MESSAGE              CHAR (40) INIT ("OUVERTURE DE
                                SESSION IMPOSSIBLE");

DCL REMOTE_SITE                 CHAR (4) INIT ("BPA6");

/* ----------------- BASED VARIABLES ------------- */

          $H_DCPMSSEMMSG PREFIX = L_CMD_
                         EVENT = CMD
                         ATTRIB = 'BASED(L_PTR_SEM)';

          $H_DCPMSSEMMSG PREFIX = L_LNK_
                         EVENT = LNK
                         ATTRIB = 'BASED(L_PTR_SEM)';

          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_CMD)'
                     PREFIX = ''
                     RECNAME = 'NBSS'
                     GEN = RECORD
                     TYPE = CMD
                     OBJ = NET;

          $H_DCPMSXR ATTRIB = 'BASED (L_PTR_REC)'
                     PREFIX = 'R_'
                     RECNAME = 'NBSS'
                     GEN = RECORD
                     TYPE = RSP
                     OBJ = NET;

          $H_DCPMSKDESC ATTRIB = 'BASED(ADDR(L_DESC))'
                        PREFIX = 'DESC_';

/* ------------------ MAIN FLOW --------------- */

          L_PTR_SEM = ADDR (L_SEMMSG);
          L_PTR_CMD = ADDR (L_COMMAND);
          L_PTR_REC = ADDR (L_RECORD);

          IF INITIATION () THEN DO;

               IF SEND_DSAC_CMD () THEN;
               CALL TERMINATION;
```

```
          END;

          ELSE
          IF EDIT_MSG (ERREUR_MESSAGE) THEN;

END_OF_EXAMPLE:
          RETURN;

/* ---------- INITIATION ------------ */

INITIATION: PROC RETURNS (BIT (1));

DCL R_RC                      BIT (16) INIT ("0000000000000000"B);

/* --- INITIATE THE CONNECTION ON REMOTE  SITE --- */

               $H_PMSOPEN SEM = ADDR(SEM1)
                          PMSID = L_PMSID
                          USERNAME = "OPERATOR"
                          PASSWORD = "OP"
                          SITE = REMOTE_SITE
                          CLEAN
                          ;

               IF $H_TESTRC ABNORMAL;
               THEN RETURN (R_FALSE);

/* - WAIT FOR THE INITIATION NOTIFICATION ON THE REMOTE SITE - */

               DO UNTIL (L_LNK_SEMMSG_RI = R_LNK_RI);
                    CALL SEPM (ADDR (SEM1), L_SEMMSG);
               END;

               IF (L_LNK_SEMMSG_RC ^= R_RC) THEN RETURN (R_FALSE);

               RETURN (R_TRUE);

          END;
/* ----- SEND_DSAC_CMD -------- */

SEND_DSAC_CMD: PROC RETURNS (BIT (1));

DCL ERR_RC                    CHAR (40) INIT ("ABNORMAL LNKRI
                              NOTIFICATION");
DCL ON_SITE                   CHAR (40) INIT ("FROM SITE : ");
DCL O_SITE                    CHAR (4);

DCL L_MAX                     FB15 INIT (1024);
DCL 1 L_MESSAGE,
      2 OUTPUT_LENGTH         FB15,
      2 OUTPUT_MESSAGE        CHAR (1024);

/* --- SEND THE COMMAND "NBSS" ON REMOTE  SITE ----- */

/* -- COMMAND IDENTIFIER --- */
               L_CMDID = "0001"X;
```

```
/* --- RESET THE PRESENCE MASK ---- */
               SUBSTR (STRING (SS_NB_CMD), 1,
               MEASURE (SEL_MASK))
               = REPEAT ("00"H, MEASURE (SEL_MASK) - 1);

/* --- SEND THE REQUEST TO THE OBJECT MANAGER -- */
               $H_PMSSENDCMD PMSID = L_PMSID
                             CMDID = L_CMDID
                             DESC = SS_NB_HD
                             DESCLN = 'MEASURE(SS_NB_HD)'
                             REC = SS_NB_CMD
                             RECLN = 'MEASURE(SS_NB_CMD)'
                             ;
               IF $H_TESTRC ABNORMAL;
               THEN RETURN (R_FALSE);
/* --- WAIT FOR THE NOTIFICATION OF "RESPONSE AVAILABLE" -- */
               DO UNTIL (L_CMD_SEMMSG_SECRI = R_CMD_SECRI);
                    CALL SEPM (ADDR (SEM1), L_SEMMSG);
                    IF L_CMD_SEMMSG_RI ^= "000"X
                    THEN DO;
                      $H_PMSCVSITE INFORM = INTERNAL
                                   SITE_IDENT = L_LNK_SEMMSG_SITE
                                   SITE_NAME = O_SITE
                                     ;
                     IF ^EDIT_MSG (ERR_RC) THEN RETURN (R_FALSE);
                     SUBSTR (ON_SITE, 16, 4) = O_SITE;
                     IF ^EDIT_MSG (ON_SITE) THEN RETURN (R_FALSE);
                         RETURN (R_FALSE);
                    END;
               END;
LOOP:
               DO UNTIL (L_STATE_STATUS = "100"B);
                    $H_PMSGETRP PMSID = L_CMD_SEMMSG_PMSID
                                CMDID = L_CMD_SEMMSG_CMDID
                                MAXDESCLN = 'MEASURE(L_DESC)'
                                DESC = L_DESC
                                DESCLN = L_DESCLN
                                MAXRECLN = 'MEASURE(L_RECORD)'
                                REC = L_RECORD
                                RECLN = L_RECLN
                                STATE = L_STATE
                                ;

                    IF $H_TESTRC ABNORMAL;
                    THEN RETURN (R_FALSE);
```

```
                        $H_PMSEDTMSG DESC = L_DESC
                                     DESCLN = L_DESCLN
                                     REC = L_RECORD
                                     RECLN = L_RECLN
                                     MESSAGE = L_MESSAGE
                                     MAXLN = L_MAX
                                     ;
                        IF $H_TESTRC ABNORMAL;
                        THEN RETURN (R_FALSE);

                        $H_PUTIOF MESSAGE = 'ADDR(OUTPUT_MESSAGE)'
                                  LENGTH = 'OUTPUT_LENGTH'
                                  ;
                        IF $H_TESTRC ABNORMAL;
                        THEN RETURN (R_FALSE);

/* END OF DO LOOP */
                END LOOP;
                RETURN (R_TRUE);
END_OF_SEND_DSAC_CMD:
           END SEND_DSAC_CMD;

EDIT_MSG: PROC (MESSAGE) RETURNS (BIT (1));
DCL MESSAGE                     CHAR (40);

                $H_PUTIOF MESSAGE = 'ADDR(MESSAGE)'
                          LENGTH = 'MEASURE(MESSAGE)'
                          ;
                IF $H_TESTRC ABNORMAL;
                THEN RETURN (R_FALSE);
                RETURN (R_TRUE);
END_OF_EDIT_MSG:
           END EDIT_MSG;

/* ------------ TERMINATION ----------------- */
TERMINATION: PROC;
                $H_PMSCLOSE PMSID = L_PMSID;
END_OF_TERMINATION:
           END TERMINATION;

END_OF_EX:
     END;
```

Glossary

This glossary explains terms and acronyms used in this manual.

A

Administrative Exchange Protocol (AEP)
Protocol used by DOF 7-PO for network administration. This protocol is used to exchange records between GCOS 7 Version 6 and a 'heterogenous' system, for example a Datanet, or a system running another version of GCOS 7. (Compare with GCOS protocol.)

C

Class of messages
A class of messages is the set of unsolicited messages matching a certain filter set.

Command
A command is a request submitted by a DOF 7-PO requestor to an object manager to maintain certain system or network object(s), or to get information about or to modify the attribute(s) of some system or network object(s).

Component
Another term for object manager.

Connection
see DOF 7-PO connection.

D

Descriptor
A descriptor is a programmatic structure describing the format manager (OMH or DSAC), the name of the DOF 7-PO object, the parameter format, the version number and the parameters of each region of the related record, if they exist.

DOF 7-PO
DOF 7-PO stands for Distributed Operator Facility - Programmed Operator. It is both a programmatic interface and a protocol to exchange administrative data between a DOF 7-PO requestor and an object manager. The object manager may be located on the same system as the requestor or on a remote system.

DOF 7-PO application
A DOF 7-PO application is a system or a user program which uses the DOF 7-PO services to monitor or to operate one or several DPS 7000 systems and/or an ISO/DSA network. Its goal is: either to control a part of the system or network operation or to make available at a terminal the function of a system or network operator. A DOF 7-PO application behaves as a DOF 7-PO requestor.

DOF 7-PO connection
The DOF 7-PO services used by the DOF 7-PO applications are connection-oriented services, so connection(s) must be initiated before using the DOF 7-PO services. A DOF 7-PO connection is created each time a H_PMSOPEN function is completed normally.

DOF 7-PO kernel
The part of DOF 7-PO responsible for data delivery and integrity. The form of the data transferred is transparent to the kernel. This aspect is handled by the format manager.

DOF 7-PO object
A DOF 7-PO object is formed of the data sent during an exchange between a DOF 7-PO requestor and an object manager, i.e., a command, a response or an unsolicited message.

DOF 7-PO queue file
A DOF 7-PO internal file used to store chains of commands and related responses, or chains of unsolicited messages.

DOF 7-PO requestor
A DOF 7-PO requestor is the initiator of a DOF 7-PO connection (that is, a DOF 7-PO application). A DOF 7-PO requestor is defined by a user identity and a DOF 7-PO connection number, and the same user may initiate up to 16 connections.

DSAC
Distributed Systems Administration and Control. The part of GCOS 7 that is responsible for network administration.

E

Edit
In DOF 7-PO, to translate the contents of a structured record into a string of characters for display on a terminal, or for printing.

Elementary response
One of a chain of responses to a command, linked together in a chain, and accessible with the H_PMSGETRP GPL primitive or C language function.

Event
An occurence or alert, for which a notification is given to the application. In DOF 7-PO, there are three events that may be signalled on the semaphore declared by the DOF 7-PO application: response available, unsolicited message available, and DOF 7-PO services (un)available.

F

Filter
A filter is a set of criteria, defining common characteristics of certain unsolicited messages (e.g. issuing job identifier, issuing RON, message key, and so on). A DOF 7-PO application can define filters to select which messages will be sent to it (see the *System Operator's Guide* for details).

Filter set
A filter set is a set of the filters used to determine whether an unsolicited message is to be sent to a certain recipient (see the *System Operator's Guide*).

Format manager
A format manager is the set of procedures used to define and handle a set of administrative data related to certain system or network objects.A format manager has two main purposes: it defines how to address the recipient of administrative data, i.e., how to invoke an object manager to execute a command or how to obtain the list of recipients for an unsolicited message. It knows the format of the exchanged administrative data and is thus able to verify the syntax, to perform a conversion or translation and to furnish the edited string in the format expected by the object manager. In the current release of DOF 7-PO, there are two format managers: OMH and DSAC.

G

GCOS protocol
This protocol is used to exchange records between GCOS 7 Version 6 and a 'homogenous' system, that is, another system running GCOS 7 Version 6. (Compare with Administrative Exchange Protocol.).

GCOS 7 structured record
The format used by the DOF 7-PO protocol to transfer data between the DOF 7-PO requestor (the application) and the object manager. It consists of a descriptor (the fixed part), and a record (the variable part).

Generic messages
Generic messages are those unsolicited messages that will be received by all main operators who have set the appropriate filters (compare with specific messages).

K

Key
Identifier of a response or an unsolicited message in a message catalog built by the message processor. The key or name of a message is a string of 4 alpha-numeric characters at maximum; the first characters of the key must be alphabetic and the last ones must be numeric, the key begins with at least 2 alphabetic characters and ends by at least 1 numeric character. Usually a key is composed of 2 alphabetic characters followed by 2 numeric characters.

L

LAEH
Local Administrative Exchange Handler. A server module that implements DOF 7-PO functions on the local system.

M

Main operator
The main operator is a human operator or an application, with the access rights to the priviledged MAIN operator commands.

Message
Strictly speaking, a message in the DOF 7-PO sense is formed of the administrative data sent during an elementary exchange between a DOF 7-PO requestor and an object manager, i.e., a command, a response or an unsolicited message. In this manual, this is referred to as a "DOF 7-PO object". An unsolicited message, which elsewhere is often called simply a "message", is written as "unsolicited message" in this manual

Modification region
The part of the GCOS 7 structured record that contains the variable fields for the new values to be established by the command.

N

Network object
An object that can be the target of one of the network commands listed in Appendix A.

O

Object
An object is a single instance of a logical or physical entity (job, output, node, line, device) in a system or network. It is represented to the software as a data structure, and often to an operator as an alphanumeric name.

Object manager
An object manager is the set of procedures used to handle a system or network object, i.e., to maintain the object, or to get or to modify the attributes of the object. A system or network object in this document is either a GCOS 7 system object or an ISO/DSA network object.

OMH
Operator Message Handler. An object manager on the local system that responds to DOF 7-PO commands.

Operator
An operator may be a person or a programmed operator. An operator may exist and be addressed apart from a DOF 7-PO connection while a DOF 7-PO requestor requires the existence of a DOF 7-PO connection.

P

Parameter
A part of a region in a GCOS 7 structured record that consists of a set of elementary fields. It specifies the options required by the command.

Presence mask
The part of the GCOS 7 structured record region that specifies whether the individual fields of that region are significant or not.

Priority
In DOF 7-PO, the enqueuing priority is the priority that the different events have for notification on the semaphore.

R

RAEH
Remote Administrative Exchange Handler. A server module that handles the exchange of DOF 7-PO objects with remote systems.

Record
A record is a programming structure used to specify the parameters of the DOF 7-PO object. It is composed of one to three regions related to the selection, modification or response parameters. Each region is usually composed of a presence mask and of an area containing the actual values of the parameters. It is defined only if the object contains variable fields.

Region
A division of the GCOS 7 structured record containing the selection, modification, or response parameters.

Repeatable message
An unsolicited message that is repeated at a definite time interval until it is answered by an operator.

Request identifier
The part of the semaphore message that identifies one of the three possible events.

Requestor
see DOF 7-PO requestor.

Response
A response is a DOF 7-PO message issued by an object manager and related to a previous command submitted by a DOF 7-PO requestor. All commands have at least one response, which may be empty.

Response region
The part of the GCOS 7 structured record that contains the response to a command, or an unsolicited message.

ROF
Remote Output Facility.

S

Selection region
The part of the GCOS 7 structured record that defines the system or network object or objects to be addressed by a command.

Specific messages
The unsolicited messages that always go directly to a specific operator (compare with generic messages).

State
In DOF 7-PO, the state of a chain of a command and its related responses, or of a chain of unsolicited messages. By testing this parameter, the DOF 7-PO application can know, for example, whether all the elements of a chain have been delivered or not.

System object
An object that can be the target of one of the system or TDS commands listed in Appendix A.

U

Unsolicited message
An unsolicited message is an administrative message issued by an object manager in consequence of some event or alert. Unlike the responses, an unsolicited message is not related to a previous command. The concepts of destination and filters are used to define their recipients. In networking, an unsolicited message is also called an unsolicited event.

Index

O

P

Q

R

S

T

U