System Call from COBOL

DPS7000/XTA
NOVASCALE 7000

Languages: COBOL

REFERENCE
47 A2 04UL 05

DPS7000/XTA NOVASCALE 7000

System Call from COBOL

Languages: COBOL

May 2002

BULL CEDOC

357 AVENUE PATTON

B.P.20845

49008 ANGERS CEDEX 01

FRANCE

REFERENCE
47 A2 04UL 05

Suggestions and criticisms concerning the form, content, and presentation of this book are invited. A form is provided at the end of this book for this purpose.

To order additional copies of this book or other Bull Technical Publications, you are invited to use the Ordering Form also provided at the end of this book.

Trademarks and Acknowledgements

Preface

Scope and Objectives

This manual describes the set of system primitives by means of which the COBOL programmer can request services from GCOS 7.

Intended Readers

This manual is intended for analysts and programmers who are familiar with the COBOL language, and who will use COBOL facilities for applications under GCOS 7.

Prerequisites

It is assumed that the reader is familiar with the COBOL language and with COBOL terminology.

Structure

Section 1:	Describes COBOL calls for Job Management.
Section 2:	Describes COBOL calls for File Management.
Section 3:	Describes COBOL calls for Time Management.
Section 4:	Describes COBOL calls for C language functions.
Section 5:	Miscellaneous calls.
Section 6:	Describes TDS calls.
Section 7:	Describes AUPI calls.
Section 8:	Describes GTWriter calls.
Section 9:	Describes UFT calls.
Appendix A:	Describes Copy Files for Catalog OUTFILEs.
Appendix B:	Describes Copy files for LIST_VOLUME or LIST_FILE OUTFILE contents
Appendix C:	Describes QUEUED files
Index	

Bibliography The following manuals describe related topics:

COBOL 85 Reference Manual	*47 A2 05UL*
GCL Programmer's Manual	*47 A2 36UJ*
COBOL 85 User's Guide	*47 A2 06UL*
System Administrator's Manual V8/V9	*47 A2 54US*
JCL User's Guide	*47 A2 12UJ*
JCL Reference Manual	*47 A2 11UJ*
Coupled Systems User's Guide	*47 A2 30UF*
GAC-Extended User's Guide	*47 A2 12UF*
MCS User's Guide	*47 A2 32UC*
C Language User's Guide	*47 A2 60UL*
TDS COBOL Programmer's Guide	*47 A2 33UT*
AUPI User's Guide	*47 A2 76UC*
GTWriter User's Guide	*47 A2 55UU*
UFT User's Guide	*47 A2 13UC*
Remote Facilities - DPS 7 to DPS 7 User's Guide	*47 A2 12UC*
Unit Record Device User's Guide	*47 A2 03UU*
DJP User's Guide	*47 A2 14UC*
Messages and Return Codes Directory.	*47 A2 10UJ*

Table of Contents

A. Copy Files For the Catalog OUTFILEs

Table of Graphics

1. Job Management

1.1 User Record Insertion in Accounting (H_ACT_UPANCT)

A user program can register accounting records of its own by using the external call to the system procedure "H_ACT_UPACNT".

DATA DESCRIPTION Statements

```
01 USER-RECORD.
   02   RECORD-TYPE PICTURE X(2).
   02   INFO PICTURE X(n).

77 INFO-LENGTH      USAGE IS COMP-1.
77 HEADER-OPTION      PICTURE X.
```

COBOL CALL Statement

```
CALL "H_ACT_UPACNT" USING
HEADER-OPTION,USER-RECORD,INFO-LENGTH.
```

Parameters

USER-RECORD	Input area containing the record type and the user accounting information. The record type is specified by the user program and must be in the range 50 to 99. The INFO area is to be filled by the user. Its length is given in the INFO-LENGTH parameter.

INFO-LENGTH	Length of INFO in bytes. It cannot exceed 1024 characters.
HEADER-OPTION	Specifies whether the standard header option is required. When this option is requested (HEADER-OPTION = 1) a standard header is inserted in the user record (its length must not be included in the length given as a parameter, which applies only to the user-supplied information). User records inserted into the accounting file as described above appear as described below when read from this file by the user billing programs.

COBOL Declaration of a User Record With No Standard Header

```
01 USER-RECORD.
   02  RECORD-TYPE    PICTURE X(2).
   02  USER-INFO      PICTURE X(n).
```

The record type is specified by the user program and must be within the range 50 to 99. The INFO area is defined by the user (length and contents).

The length of INFO cannot exceed 1024 characters.

COBOL Declaration of a User Record With Standard Header

```
01   USER-RECORD.
   02   RECORD-TYPE         PICTURE X(2).
   02   HEADER              PICTURE X(12)
   03   USER-NAME           PICTURE X(12).
   03   PROJECT             PICTURE X(12).
   03   BILLING             PICTURE X(12).
   03   JOBID               PICTURE X(8).
   03   RON                 PICTURE X(4).
   03   REPEATED-JOB        PICTURE X.
   03   DSN                 PICTURE X(3).
   03   DATE                PICTURE X(6).
   03   TIME                PICTURE X(6).
   03   HEADER-FLAG         PICTURE X(2).
   02   USER-INFO           PICTURE X(n).
```

Length of the record: (68 + USER_INFO length) bytes

NOTE:

DATE and TIME are those of the record insertion in the accounting file.

HEADER-FLAG is a two-character field which delimits the end of the standard header and is used by the EDITACT utility and other accounting procedures to detect whether a standard header has been inserted.

The value of HEADER-FLAG is "!!" (hexadecimal "5A5A").

1.2 User JCL Status (H_CBL_USETST)

The system sets a status value, which can be used in a JUMP JCL statement, or tested from the #status system variable in GCL, in the event of an abnormal step termination (STATUS=10000), in an operator-requested end of step (STATUS=50000) or in case of system crash (STATUS=61000). The COBOL compiler also sets the status value at the end of compilation, according to errors detected (see *COBOL 85 User's Guide,* Chapter 2, "The Compiler").

The user may also set the status value in his COBOL program, transmitting it to the run-time package routine H_CBL_USETST via a field described in the Working-Storage Section with usage COMP-1. Since COMP-1 is a binary half-word, the user status value has a limit of 32767.

Each value for STATUS has a corresponding value for the step severity code (SEV). The following table gives the correspondence between SEV and STATUS:

STATUS	**SEV**
0-99	0
100-999	1
1000-9999	2
10000-19999	3
20000-32767	4
50000	5
> 60000	6

The following example shows how the status value can be set in a COBOL program:

```
.
WORKING-STORAGE SECTION.
01 STATE COMP-1.
.
PROCEDURE DIVISION.
.
MOVE 64 TO STATE.
CALL "H_CBL_USETST" USING STATE.
 .
```

Execution of the job stream can then be modified by testing this status value:

```
$JOB...
.
STEP TEST01, TEMP, DUMP=DATA;
ENDSTEP;
JUMP LAB1,STATUS,EQ,64;
SEND ’STATUS DIFFERENT FROM 64’;
JUMP LAB2;
LAB1: SEND ’STATUS = 64’;
LAB2: SEND ’END OF TEST’;
$ENDJOB;
```

The JOR will then show:

```
PROCESS GROUP TERMINATED STATUS = 64
```

if the CALL statement is executed.

1.3 Checkpoint, Restart and Journalization (H_CK_UCHKPT, H_CK_UMODE)

The RERUN clause in the I-O-CONTROL paragraph allows the user to specify the conditions, if any, under which checkpoints are to be taken during program execution. Checkpoints can be taken at each end of volume in a specified file or each time a specified number of records is read or written in a specified file. The checkpoint may be somewhat delayed depending on I/O events; usually, no checkpoint is taken for an I/O operation that does not return a "00" status.

Checkpoint data are placed in Backing Store. If the program aborts or if there is a system crash, and the STEP statement contains the REPEAT parameter, the operator may call for the program execution to be restarted. If he does so, the program is restored to its state at the last checkpoint and execution continues from there. The REPEAT parameter of the $JOB statement can be used to request the restart of an entire job.

The user can also request checkpoints in the execution JCL. See the DEFINE JCL statement in the *JCL Reference Manual*.

At the price of introducing a non-standard element into his source program, the user may also directly call the system checkpoint procedure H_CK_UCHKPT, giving two parameters. For example:

```
CALL "H_CK_UCHKPT" USING RMODE, INFO.
```

RMODE is a user-defined USAGE COMP-2 field which indicates whether the current execution of the program is the first execution (RMODE = zero) or if the program has been restarted (RMODE not = zero). In the latter case, RMODE contains the JCL status value for the abnormal step termination, which also appears in the JOR.

INFO is a user-defined group item consisting of 32 one-character elements. Each character of the returned string is either "0" or "1" and indicates the occurrence of a given condition when set to "1". If [1] refers to the leftmost character and [32] to the rightmost character of the string, then the meaning of each flag is as follow:

- [5] the process-group is multi-process.
- [6] the DEBUG parameter was specified in the STEP statement. Checkpoint is not taken.
- [7] the REPEAT parameter was not defined in the STEP statement.
- [14] the checkpoint failed. Refer to the JOR for the reason.
- [16] major error. Refer to the JOR.
- [32] the next checkpoint will not be taken.

Regardless of whether a checkpoint is taken as a result of the RERUN clause or a programmed CALL, these values can be checked by coding:

```
CALL "H_CK_UMODE" USING RMODE INFO.
```

where RMODE and INFO have the same meaning as for H_CK_UCHKPT. This CALL also introduces a non-standard element into the user's source program, and will require alteration to run on any other system.

Associated with checkpointing is "journalization". This is a facility offered by Data Management which keeps a record of all file updates so that files can be reconstituted before a rerun is performed.

More information on the use of the above facilities are given in the *System Administrator's Manual* and the *JCL User's Guide*.

1.4 Commitment Call (H_GAC_UCOMIT)

The following COBOL declarations are required:

```
77         MODE         COMP-2.
77         CKINF        PIC X (32).
77         NUMLOCK      COMP-1 VALUE -1.
77         NOCHKPT      PIC X VALUE SPACE.
77         CURRENCY     PIC X.
```

The COBOL call has the following format:

```
CALL "H_GAC_UCOMIT" USING      MODE,
                               CKINF,
                               NUMLOCK,
                               NOCHKPT,
                               CURRENCY
```

Description of Parameters

MODE	A numeric variable which specifies whether the current execution mode is normal or restart after an incident. If the execution mode is normal, MODE is set to zero. If the execution mode is restart, MODE is set to the value of the step completion code at the time of the incident.
CKINF	This parameter is a 32-character output string. Depending on certain conditions, some of the characters in the string may be set to 1. Use [1] to indicate the leftmost character in the string, and [32] the rightmost character, then: [6] = 1: DEBUG has been specified in the $STEP statement. Checkpoints are not taken. [14] = 1: The checkpoint has failed. See the Job Occurrence Report. [16] = 1: A major error has occurred. See the Job Occurrence Report. [32] = 1: The next checkpoint will not be taken.
NUMLOCK	This parameter is no longer taken into account. It is retained only for compatibility with previous releases.

NOCHKPT	This parameter can be either blank or non-blank. If GAC-EXTENDED is fully effective for the step (there is at least one file open for which locks may be applied), it is ignored. It is meaningful only if GAC-EXTENDED is NOT fully effective for the step. In this case, when NOCHKPT is blank, commitment calls are processed as checkpoints, but if NOCHKPT has any other value, commitment call is ignored. This feature allows programs to be run without alteration whether the files they use are controlled by GAC-EXTENDED or not.
CURRENCY	This parameter specifies whether the current record pointers are lost or retained after a commitment has been taken. CURRENCY is either L (lose pointers) or K (keep pointers). If CURRENCY has another value, LOCKMARK parameter of DEFINE JCL applies. If LOCKMARK is specified for a given file, the current record pointer will be retained at the end of the commitment unit. If it is not specified, the current record pointer will be lost.

1.5 JOB SUBMISSION

1.5.1 H_IN_ISUBMIT

COBOL Syntax

```
CALL "H_IN_ISUBMIT" USING job-description, status, file-description, interface.
```

Description

Submits a request for asynchronous job execution. A submitted job is stored in the Stream Reader queue and processed by the Stream Reader service asynchronously. The term asynchronous means that the program submitting the request can continue to do its own work while the submitted job becomes eligible for execution.

A job submission and its execution are asynchronous. The console and report messages are not directed to the submitting terminal (the user specified in the $JOB statement), but to the IOF mailbox of the user submitting the program which contains the CALL "H_IN_ISUBMIT" procedure.

The user can submit a job to a remote host and direct its output to a different destination.

Note the following points:

- $JOB/$ENDJOB present in the JCL of the submitted job:

  You must specify the PROJECT and BILLING parameters in the $JOB statement of the submitted job if there are no corresponding default values in the site catalog; otherwise the submitted job aborts.

- No $JOB/$ENJOB present in the JCL of the submitted job:

  The project and billing of the user submitting the program which contains the CALL "H_IN_ISUBMIT" statement is used.

For more information, see the *JCL Reference Manual*.

Usage

Job-description is a data structure that contains the set of parameters applicable to the submitted jobs. The job description must be structured as follows:

```
01 JOB-DESCRIPTION.
 02 JOBDESC-STRUCT.
    03 JOBDESC-STRUCT-LN          COMP-1 VALUE 61.  ]
    03 JOB-CLASS                  PIC X.            ]
    03 JOB-PRIORITY               PIC 9.            ]
    03 JOB-SWITCHES               PIC X   OCCURS 32.]
   03 JOB-DELETE                 PIC X.             ] 61 bytes
    03 JOB-HOST                   PIC X(4).         ]
    03 JOB-CLASS2                 PIC X(2).         ]
    03 JOB-SKIP-BLANK             PIC X.            ]
    03 FILLER                     PIC X.            ]
    03 JOB-OUTDEST.                                 ]
       04  PRIMARY-DEST           PIC X(8).         ]
       04  SECONDARY-DEST         PIC X(8).         ]
 02 JOB-VALUES.
    03 VALUES-STRUCT-LN           COMP-1.
    03 VALUES-STRUCT.
       04 VALUES-STRUCT-HEADER.
          05 NB-OF-POSITIONAL     COMP-1.
          05 NB-OF-KEYWORD        COMP-1.
       04 VALUES-PARAMETERS       PIC X(m).
```

All the fields in JOB-DESCRIPTION are input parameters.

JOBDESC-STRUCT-LN	Must contain the exact size in bytes of the JOB-DESCRIPTION structure. This value is typically 61.
JOB-CLASS	If specified, must be one of the sixteen classes from A to P under which the job will be submitted, scheduled and executed. If JOB-CLASS contains a class which is not available to the recipient of the submitted job, the class of the submitted job will be P. If JOB-CLASS is filled with spaces, refer to JOB-CLASS2.
JOB-PRIORITY	Represents the job scheduling priority such as can be specified in the $JOB statement. It must be in the range 0 to 7, where 0 is the highest and 7 is the lowest priority.

JOB-SWITCHES	Represent the initial job switch values. You must initialize each of them to 0 or 1. They are numbered from 1 to 32 whereas the switches specified in the SWITCHES parameter of the ENTER_JOB_REQUEST directive are numbered from 0 to 31.
JOB-DELETE	Must be set to Y (Yes) or N (No). If yes, the subfile which contains the job to be submitted will be deleted once the job has successfully executed. This parameter corresponds to the DELETE option of the $JOB statement.
JOB-HOST	Is a DSA node name up to 4 characters long. It specifies the remote host name where the job must be executed. This parameter corresponds to the HOST option of the $JOB statement. If equal to spaces, the job is executed locally. The host name must be cataloged.
JOB-CLASS2	Allows you to specify 2 characters for the class of the job to be submitted. If JOB-CLASS contains a value other than a space, then JOB-CLASS2 must be filled with spaces. If JOB-CLASS2 contains a value other than spaces, then JOB-CLASS must contain a space. Both JOB-CLASS and JOB-CLASS2 can be filled with spaces in which case the class of the submitted job is the default batch class for the project. If JOB-CLASS2 contains a class which is not available to the recipient of the submitted job, the class of the submitted job will be P.
JOB-SKIP-BLANK	Must be set to Y (yes) or N (no). If yes, spaces to the right of the passed valued are suppressed. If no or any other value, spaces are kept. (Note that this was the default value in V5.)

JOB-OUTDEST	The two fields of this substructure specify the output destination station. These parameters correspond to the DEST option of the JCL statements SYSOUT, WRITER, or OUTVAL. If both are blank, the job output is directed to the local main station. If the primary destination name is specified (not blank) and the secondary destination name is not initialized, then the job output is directed to the primary RBF station. If the secondary destination name is also initialized, then the output is directed according to the Distributed Job Processing algorithm.

The JOB-VALUES substructure contains the initial job values.

VALUES-STRUCT-LN	Defines the size in bytes of the VALUES-STRUCT data structure. If 0, there is no value to be transmitted to the job.
NB-OF-POSITIONAL	Defines the number of positional values which are described in the values parameter structure.
NB-OF-KEYWORD	Defines the number of keyword values which are described in the values parameter structure. When the submitted job is a GCL procedure NB-OF-KEYWORD is equal to zero.
VALUES-PARAMETERS	Defines all the elements (positional and keyword values). This structure is the concatenation of all the elements. Each element must have one of the following descriptions. If the element is a positional value, then the description is:

```
05 POSITIONAL.
06 POS-LENGTH COMP-1.
06 POS-VALUE PIC X(POS-LENGTH).
```

Where, POS-LENGTH is the size in bytes of the positional value.

If the element is a keyword value, then the declaration is as follows:

```
05 KEYWORD.
06 KW-LENGTH COMP-1.
06 KW-NAME   PIC X(8).
06 KW-VALUE  PIC X(KW-LENGTH).
```

where KW-LENGTH is the size in bytes of the keyword values and KW-NAME is the keyword name left justified and padded with spaces.

All the positional values must be declared before the keyword values.

If the default job value is chosen, you must specify the following for:

– a positional value:
```
   06  POS-LENGTH COMP-1 VALUE 0.
   (do not use POS-VALUE)
```

– a keyword value:
```
   06  KW-LENGTH COMP-1 VALUE 0.
   06  KW-NAME PIC X(8).
   (do not use KW-VALUE)
```

If the positional value or keyword value are not completely filled (that is they contain spaces on the right), the spaces are passed to the reader and can lead to JCL translation errors.

This can be seen in the following example:

```
05 POSITIONAL .
  06 POS1_LGTH  COMP-1 VALUE 6.
  06 POS1_VL PIC X(6) VALUE "ABC  ".
```

In the example above, if POS1_VL is used in:

```
   MVL A=&l_XYZ;
```

Then, the value of A is ABCbbb_XYZ and not ABC_XYZ.

When the submitted job is a GCL procedure the first POS-VALUE must contain:

```
   MWINLIB BIN binary-library-name
              [:media:device-class];
```

Library name is the name of the library where the GCL procedure is stored.

The second POS-VALUE must contain:

```
procedure-name b
```

where:

`procedure-name` is the name of the GCL procedure
b represents the blank character suffix which you must add on to the procedure name.

The third and possibly the fourth POS-VALUE fields must contain the parameters of the GCL procedure. Each POS-VALUE length is limited to 128 characters.

- Status is a data structure which defines the status of the CALL "H_IN_ISUBMIT" statement. It is an output parameter and must have the following data structure:

```
  01 SUBMIT-STATUS.
     02  RESULT            PIC 9.
     02  ERROR-TYPE            COMP-1.
     02  ERROR-NB              COMP-1.
  01 SUBMIT-STATUSB REDEFINES SUBMIT-STATUS.
     02  RESULTB           PIC 9.
     02  REQID                 COMP-2.
```

NOTE:
RESULTB is a another name of RESULT (Redefines RESULT).

RESULT = 0	Successful completion. A request to submit the job is made. The REQID field of the STATUS structure is filled with the Request Identifier and can be used to get information on the submitted job (see the paragraph CALL "H_CBK_UJOBINFO" below).
RESULT = 1	Abnormal completion. In this case, ERROR-TYPE = 1 to indicate an error.
ERROR-TYPE	Value 0 if RESULT is also 0 Value 1 to indicate an error.

ERROR-NB	Gives the reason for the error. Possible values are: "2" - wrong priority (JOB-PRIORITY) "4" - wrong class (JOB-CLASS) "13" - error in the values (JOB-VALUE) "26" - wrong switch(es) (JOB-SWITCHES) "27" - wrong subfile suppression value (JOB-DELETE) "29" - wrong syntax (OCL or GCL) in the file description (FILE) "30" - correct syntax for the parameters but failure of job submission (for example JOB_HOST correct syntax but non-existent) "33" - error in the site name (for example incompatibility between the local system and the remote host at which the file resides)
REQID	Internal Request Identifier. This is the REQID that is used to obtain information about the submitted job.

- `File description` is a data structure which identifies the "file literal" description of the file which contains the job to be submitted.

It must have the following format:

```
01 FILE-DESCRIPTION.
   02  FILE-LITERAL-LENGTH  COMP-1.
   02  FILE-LITERAL         PIC X(FILE-LITERAL-LENGTH).
```

where, `FILE-LITERAL-LENGTH` is the size in bytes of the `FILE-LITERAL` item.

In **OCL** (Operator Control Language), `FILE-LITERAL` has the following format:

```
[subfile-name:] external-filename [:media:device-class]
```

or

In **GCL** (GCOS Command Language), `FILE-LITERAL` has the following format:

```
external-filename [..subfile-name] [:media:device-class]
```

When the submitted job is a GCL procedure, the format is as follows:

```
SYS.HSLLIB..ABSENTEE
```

When the file is cataloged, it must be cataloged in the site catalog or a private auto-attachable catalog.

Media and device class identify respectively the volume and device on which the external filename resides. They must not be specified when the file is cataloged.

The file may be either a sequential file or a library member.

Interface is a data structure specifying a COBOL call.

It is an input parameter and must have the following structure and contents:

```
01 STRUCT-INTERFACE.
   02 FILLER     COMP-1     VALUE ZERO.
   02 FILLER     COMP-1     VALUE -1.
```

1.5.2 H_IN_URUN

Submits a request for synchronous JCL or GCL batch job execution. For more information about H_IN_URUN, refer to the *GCL Programmer's Manual*.

1.5.3 H_IN_UEJR

Submits a request for asynchronous JCL or GCL batch job execution. For more information about H_IN_UEJR, refer to the *GCL Programmer's Manual.*

1.5.4 H_IN_UJDERR

Generates error messages from error numbers and classes (to be used after H_IN_URUN or H_IN_UEJR). For more information about H_IN_UJDERR, refer to the *GCL Programmer's Manual*.

1.5.5 H_CBL_UJOBINFO

COBOL Syntax

```
CALL "H_CBL_UJOBINFO" USING data-name.
```

Description:

Allows to know the state of a job submitted by the call "H_IN_ISUBMIT" or "H_IN_UEJR". This procedure does not work if the submitted job is executed on a remote host, or contains a data enclosure ($DATA statement). The job outputs must be held (using the HOLDOUT keyword in JCL, or the HO command in GCL) in order to be still known by the system after the job termination.

Only one job can be asynchronously submitted. If several jobs are submitted, they have all the same REQID, so the result has no signification.

This procedure takes as input the Request Identified of the submitted job (REQID field of the STATUS structure when the call "H_IN_ISUBMIT" statement is successful), and returns as output the state of the job.

The CALL "H_CBL_UJOBINFO" when it's used, follows a CALL "H_IN_ISUBMIT" or a CALL "H_IN_UEJR".

Usage:

Data-name is a structure with the following format:

```
01 JOB_STRUCT.
   02 REQUID        COMP-2.
   02  RESULT       COMP-1.
   02  RC           COMP-2.
   02  RON          COMP-1.
   02  JOBSTATE     COMP-1.
   02  SUBJOBSTATE  COMP-1
```

REQID is an input parameter, that contains the Request Identifier returned by H_IN_ISUBMIT or by H_IN_UEJR

RESULT gives the result of the "JOBINFO" procedure call.

- 0 - successful completion of the procedure
- 1 - unsuccessful completion of the procedure.

RC can be edited in the following way:

```
01 EDITRC     PIC X (30).
   CALL "H_STD_UEDTG4" USING EDITRC ADDRESS OF RC.
   DISPLAY "RETURN CODE="EDITRC.
```

RC can take the following values:

DONE	the function JOBINFO is completed (RESULT = 0)
ARGERR	invalid argument REQID.
NOMATCH	the JOB is no more known by the system or it's not yet introduced.
SYSOVLD	overflow on a system table.

RON is the binary RON of the submitted Job identified by REQID (when RESULT = 0, otherwise ROB = 0).

JOBSTATE can take the following values:

- 0 - UNKNOWN (in this case RESULT = 1)
- 1 - IN INTRODUCTION
- 2 - READ
- 3 - IN TRANSLATION
- 4 - TRANSLATED WITH ERROR
- 5 - HELD
- 6 - SCHEDULABLE
- 7 - IN EXECUTION
- 8 - SUSPENDED
- 9 - TERMINATED
- 10 - IDLE
- 11 - INTRODUCED WAITING INPUT - or it’s no yet introduced

SUBJOBSTATE is a precision on the JOB termination when JOBSTATE has the value "TERMINATED"(otherwise SUBJOBSTATE = 0).

SUBJOBSTATE can take one of the following values:

- 0 - JOB COMPLETED
- 1 - USAGE OF $DATA   Input Reader statement=$DATA
- 2 - NOT EXECUTED
- 3 - ABORTED
- 4 - JOB has been killed with TJ strong
- 5 - JOB has been killed with TJ weak

A JOB is completed when JOBSTATE = 9 and SUBJOBSTATE = 0

❑

2. File Management

2.1 Locking and Unlocking of Files (H_DFPRE_CLKF)

2.1.1 Introduction

The techniques of file reservation use locks stored in the file labels. These locks are set when a file is assigned using the primitive H_LOCK, and reset when the file is de-assigned, using the primitive H_UNLOCK. For multi-volume files they are stored only in the shared volume which contains the lowest Volume Sequence Number for the file.

H_LOCK and H_UNLOCK are available to users working in COBOL or GPL for files for which the SHARE parameter (given either in the catalog entry for the file, or when the file is assigned) is either ONEWRITE or FREE. They may be used in a coupled systems or in a single system environment.

These primitives do not apply for TDS controlled files. Moreover, when a controlled file is open, the primitives are ineffective.

2.1.2 Restrictions on the Use of H_LOCK and H_UNLOCK

The primitives H_LOCK and H_UNLOCK apply for all file organizations. However, users wishing to use these primitives must bear in mind the following restrictions on their use:

- H_LOCK and H_UNLOCK cannot be used for TDS controlled files, nor for files controlled by the General Access Control facility (GAC).
- Before H_LOCK and H_UNLOCK can be used to share files, the files must have been assigned, using the JCL statement ASSIGN or the GCL parameter group ASG(i), and allocated, using the JCL statement ALLOCATE, or the GCL command BUILD_FILE.

- Files cannot be shared using H_LOCK and H_UNLOCK if any other files in the same step are TDS controlled files, or files controlled by GAC. If GAC is active for a step which calls H_LOCK, the return code SHCTVIOL or INVUSE is output, and the primitive call fails.
- If checkpoints are being taken, a file must be locked and released between two checkpoints. If a checkpoint is taken between the call to H_LOCK and the call to H_UNLOCK, exclusive control of the file is lost when the step is restarted after the checkpoint.

2.1.3 Locking Files

The primitive H_LOCK gives exclusive control of a shared file to the task which calls it.

2.1.3.1 Calling H_LOCK From a COBOL Program

The call to H_LOCK from a COBOL program is the following:

```
CALL "H_DFPRE_CLKF" USING  file-name, time-slice, repeat, request
```

where:

file-name	specifies the COBOL file name to be locked.
time-slice	(COMP-1) specifies a time interval measured in elapsed seconds. Following an unsuccessful attempt to lock a file, another attempt will be made after the given time interval. The default value is zero. **This parameter can only be used in a coupled systems environment.** In a single system environment, if the lock is not successful, the step is put into the WAIT state. In this case, the parameter must be specified as zero.
repeat	(COMP-1) specifies the maximum number of unsuccessful attempts that may be made to lock the file. The default value is zero. **This parameter can only be used in a coupled systems environment.** In a single system environment the parameter must be specified as zero.
request	(PIC X) must be given the value "P" to request file locking.

2.1.3.2 Return Codes

The return codes that may result from the use of this primitive are:

Normal

DONE	H_LOCK completed successfully.
ALREADY	The file has already been locked by the job step which called H_LOCK.

Abnormal

IFNERR	The internal file name has no file description. This may happen when an attempt has been made to lock a file before it has been opened. Therefore no checking has been done on the correspondence between the internal file name and the file description.
INVUSE	A request has been made to lock a file when GAC has been requested for the issuing step. Locking is rejected, to prevent deadlock.
FILENASG	The internal file name has not been assigned.
FLABUNKN	The file label does not exist.
FUNCNAV	An illegal request has been made for exclusive control of the file SYS.IN, SYS.OUT or SYS.URCINIT.
SHCTVIOL	An illegal request has been made to lock a file while GAC is active for the issuing step. Locking is rejected, to prevent deadlock.
TIMEOUT	The system has unsuccessfully tried to lock the file the number of times specified by REPEAT at intervals specified by TIMESLICE.
ARGERR	There is an error in the parameter list of the COBOL call.

NOTE:
Refer to Section 5.1, *Get or Edit a Return Code*.

2.1.4 Unlocking Files

The primitive H_UNLOCK releases exclusive control of a shared file. The lock on a file is released when H_UNLOCK terminates successfully, when the file is de-assigned, or when the step terminates.

2.1.4.1 Calling H_UNLOCK From a COBOL Program

The call to H_UNLOCK from a COBOL program is the following:

```
CALL "H_DFPRE_CLKF" USING  file-name, time-slice, repeat, request
```

where:

file-name	specifies the COBOL file name to be unlocked.
time-slice	(COMP-1) time-slice and repeat must both be zero, or both must be non-zero. Same meaning than for H_LOCK.
repeat	(COMP-1) repeat and time-slice must both be zero, or both must be non-zero. Same meaning than for H_LOCK.
request	(PIC X) must be given the value "V" to request that the file be unlocked.

2.1.4.2 Return Codes

The return codes that may result from the use of this primitive are:

Normal

DONE	H_UNLOCK completed successfully.

Abnormal

NOLOCK	The job step which called H_UNLOCK is not the step which originally locked the file.
IFNERR	The internal file name does not identify a valid file.

NOTE:
Refer to Section 5.1, *Get or Edit a Return Code*.

2.2 Overriding the Standard Parameters of a SYSOUT File (H_OW_USYSOUT)

Description

This routine can be used to dynamically override the standard parameters of a SYSOUT file, similar in effect to the JCL statement SYSOUT. These standard parameters may be the Output Writer default values, or the values specified via the OUTVAL parameter of the JCL statement $ JOB. If H_OW_USYSOUT is used and the JCL statement SYSOUT is present as well, the Output Writer will create two versions of the output file, one with the characteristics specified via H_OW_USYSOUT, and one with the characteristics specified via the JCL statement SYSOUT.

The use of H_OW_USYSOUT enables several versions of a standard SYSOUT file to be created (each with different editing parameters) from a single file description.

Format of H_OW_USYSOUT Call

The routine is called as follows:

```
CALL "H_OW_USYSOUT" USING file-name, editing-parameters.
```

where:

- file-name is the name of the file. If this file has not been declared with the SYSOUT option in the SELECT clause (of COBOL), this option will now be forced.
- editing-parameters is the name of a structure containing the editing parameters. An example of such a structure is as follows:

```
01   EDITING-PARAMETERS.
  02      NAME        PIC X (8)       VALUE SPACE.
  02      CLASS       PIC X           VALUE SPACE.
  02      PRIORITY    PIC X           VALUE SPACE.
  02      DESTINATION.
    03    STATION1    PIC X (8)       VALUE SPACE.
    03    STATION2    PIC X (8)       VALUE SPACE.
  02      HOLD        PIC X           VALUE SPACE.
  02      WHEN        PIC X           VALUE SPACE.
  02      INTERVAL    PIC 9 (4)       VALUE ZERO.
  02      COPIES      PIC 99          VALUE ZERO.
  02      BANINF.
```

```
    03    BAN1        PIC X (12)      VALUE SPACE.
    03    BAN2        PIC X (12)      VALUE SPACE.
    03    BAN3        PIC X (12)      VALUE SPACE.
    03    BAN4        PIC X (12)      VALUE SPACE.
  02      DEVCLASS    PIC X (16)      VALUE SPACE.
  02      MEDIA       PIC X (6)       VALUE SPACE.
  02      SLEW        PIC X           VALUE SPACE.
  02      DELETE      PIC X           VALUE SPACE.
  02      BANNER      PIC X           VALUE SPACE.
  02      TERMINATION PIC X           VALUE ".".
```

The description of the parameters is as follows:

NAME	Name of the output file
CLASS	Output class (a letter, A to Z)
PRIORITY	Priority (a digit, 0 to 7)
DESTINATION	Destination of the output in the RBF DJP environments. (Refer to the *Remote Facilities - DPS 7 to DPS 7 User Guide*).
HOLD	If "Y" is specified, the file is placed in the output queue at the time specified by WHEN, but is given HOLD status. An "RO" operator command is needed to release the file for printing.
WHEN	This specifies when the file is placed in the output queue. The possible values are: "J" = at the end of the job (JOB) "S" = at the end of the step (STEP) "I" = at the closing of the file (IMMED) "D" = not placed in the output queue (DEFER)
INTERVAL	This is the size (in number of pages) of an early delivery. If INTERVAL has the value 50 (for example) then this is equivalent to specify WHEN = 50 in the JCL statement SYSOUT.
COPIES	Number of copies (between 1 and 10).
BANINF	From 1 to 4 words which will replace the standard names (RON, USER, JOBID, BILLING) in the output banner.
DEVCLASS	Device class for the output.

MEDIA	Volume name for the output. For a printer, the BELT and PAPER parameters are given (see the *Unit Record Device User's Guide*).
SLEW	"N" means that the output is printed with only 1 line skip.
DELETE	"Y" means that if the output file is a member of a library file, the member is deleted after it is printed.
BANNER	This parameter may take the following values: – “N”, suppresses banners in the output (NO). – “J”, banners are printed between each job in a output (JOB). – “O”, banners are printed between each output (OUTPUT). – “C”, banners are printed between each copy of an output (COPY). – “M”, banners are printed between each member of an output (MEMBER).
TERMINATION	Terminal character, must be a dot (.).

2.3 Converting to "BIG" Characters

Description

This routine converts a user-specified text to "big" characters. These "big" characters are the same size as the characters printed by the Output Writer in the standard output banners. The converted text can be written to a file or placed in a user-specified area.

Format of H_OW_UBIG Call

This routine is called by:

```
                         {output-file}
CALL "H_OW_UBIG" USING  {            } user-text [code]
                         { user-area }
```

The description of the parameters is as follows:

output-file	This is the file-name (in the FD) of the file in which the big characters will be written, on 10 lines, as follows: – one blank line – 8 lines containing the character images which form the big characters (left aligned) – one blank line

user-area	This is the name of the area into which the converted text is to be placed. This can be specified as an alternative to specifying an output file.
	The user area has the following structure:
	`01 USER AREA.`
	`   02 LINE-LENGTH COMP2 VALUE IS nnn.`
	`   02 USER-LINE PIC X(nnn)`
	`              OCCURS 8 TIMES.`
	Where, nnn is the line length (in normal characters).
	The converted text character images are placed to the left of each line and rightmost unused positions are set to SPACES.

user-text	This is the name of the data item containing the text to be converted to "big" characters. It can contain up to 16 characters and trailing (i.e., rightmost) non-printable characters (including spaces) are ignored. If the converted text is to be written directly to a file (i.e., output-file is specified) the user must ensure that the file's record layout can accommodate the converted character images. If the converted text is to be placed in a user area (i.e., user-area is specified) the text will be truncated on the right (if necessary) to make it fit the area specified.
code	This parameter provides a means of specifying the "black" (i.e., printed) characters and the "blank" (i.e., filler) characters to be used in the conversion. If this parameter is omitted, "@" is used for each black character and " " is used for each blank character. If specified, code has the following structure:

```
01    USER-CODE.
   02         BLANK-CHARACTER  PIC X.
   02         BLACK-CHARACTER  PIC X.
```

2.4 Invalidating Updates to Files

Description

The H_JAP_HINVFRU (INValidate File Recovery Unit) primitive allows you to invalidate all modifications to all files journalized in the Before Journal. It invalidates all modifications made since the last checkpoint or since the beginning of the step (of a batch or IOF program). The primitive functions whether the files are being monitored by GAC or not, whether checkpoints are being taken or not, and whether the step is repeatable or not.

This primitive can only be called when all the files journalized in the Before Journal are closed. The CIs are rolled back and the corresponding After Images are invalidated. To continue the treatment of these files, the user must re-open them. All the CIs locked since the last checkpoint remain locked until the next checkpoint.

H_JAP_HINVFRU applies to all the journalized files, that is, UFAS and IDS/II files in processing modes that allow journalization. Refer to the *File Recovery Facilities User's Guide* (Chapter 2), for more information.

These files must be journalized in the Before Journal.

For the sake of integrity, you are strongly advised to journalize them in the After Journal, as it may be necessary to roll forward (if a rollback is impossible due to faulty media).

However, H_JAP_HINVFRU does not apply to files that are journalized only in the After Journal. This is because, in batch/IOF processing, updates are made in "immediate update" mode and only a static rollforward can re-establish the integrity of such files.

Files journalized as AFTER and not rolled forward are inconsistent with files journalized as BEFORE or as BOTH. The H_JAP_HINVFRU primitive is rejected if a journalized file is not protected by the Before Journal.

Format of H_JAP_HINVFRU Call

This routine is called by:

```
CALL "H_JAP_HINVFRU" USING  status
```

The description of the parameter is as follows:

status	This is a field (PIC X) to receive the return code (status) of the execution of the primitive. The values of the return code are listed below. There are no other parameters.

Normal Return Code Value

0	H_JAP_HINVFRU has successfully completed.

Abnormal Return Codes Values

1	H_JAP_HINVFRU was not executed because at least one file journalized in the Before Journal is open.
2	H_JAP_HINVFRU was not executed because the step is not a batch or IOF step.
3	H_JAP_HINVFRU was not executed because at least one file journalized in the After Journal is not journalized in the Before Journal.
x	Any other non-zero value indicates that H_JAP_HINVFRU was not able to execute successfully.

Constraints

For H_JAP_HINVFRU to execute correctly:

- All journalized files must be closed.
- All journalized files must be at least written to the Before Journal.
- You cannot invalidate updates for a single file. The atomicity of the file recovery unit imposes this integrity constraint.
- The primitive can be called only from a batch or IOF step.

You cannot invalidate updates for a single file. The atomicity of the file recovery unit imposes this integrity constraint.

❑

3. Time Management

3.1 Set Timer (H_TM_USETTM)

COBOL call syntax:

```
CALL "H_TM_USETTM" USING user-specified-time
```

Description

Request to put the issuing task into the waiting state for a user specified time, before being re-activated.

Usage

- user-specified-time is a COMP-2 input field specifying, in milliseconds, how long the task has to be suspended.
- H_TM_USETTM is mainly used for optimizing MCS applications (Refer to the *MCS User's Guide*).
- H_TM_USETTM should not be used with TDS as it can lead to TIME-OUT and transaction abort.

Return codes

Normal

DONE	The task has been put into the waiting state.

Abnormal

ENTRYOV	The request cannot be satisfied immediately, try later.

NOTE:
Refer to Section 5.1 *Get or edit a return code*.

❑

4. C Language

4.1 C Run Time Package

A C program consists of a set of functions, one of which is known as the main function and called first. The main function automatically calls the C Run Time package before the first executable statement in order to initialize its own working area. In the same way, the main function calls the C Run Time package after its last executable statement, in order to flush buffers and close files (except when an explicit return statement is executed).

Nevertheless, it is possible to initialize the C Run Time package without going through the main function, by invoking explicitly the H_CLR_EPROLOG function. The H_CLR_EPILOG function must be used to clear the Run Time working area at the end of the run unit.

The respective prototypes are:

- extern void H_CLR_EPROLOG ();
- extern void H_CLR_EPILOG ();

Subroutines, written in other languages supported by GCOS 7, can call these functions.

4.2 Run Time Initialization from a COBOL Program Not in a TPR (h_clr_eprolog, h_clr_epilog)

A run unit can contain COBOL programs calling C functions.

In such a run unit the C functions must not call the main function and the C run time initialization can be done in a COBOL program before the first call of the C function by:

```
CALL "H_CLR_EPROLOG"
```

After the last call of a C function, and before the end of the run unit, the C run time working area must be cleared, possibly from a COBOL program by:

```
CALL "H_CLR_EPILOG"
```

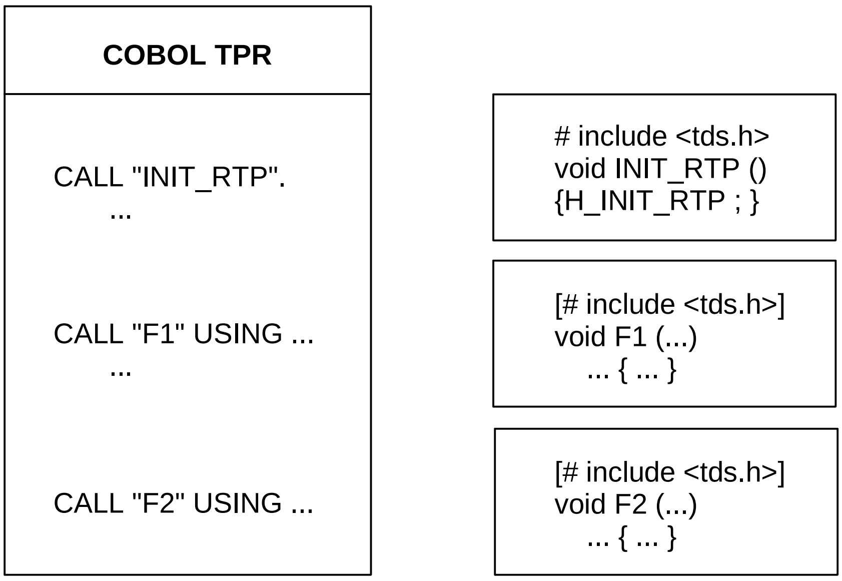
4.3 How to Call a C Function in a COBOL TPR (H_INIT_RTP)

In order to call a C function in a COBOL TPR, the COBOL TPR must contain a call on the special C function "INIT_RTP". This call must come before the first call of a function in the COBOL TPR.

This is shown as follows:

```
#include <tds.h>
void INIT_RTP()
  {H_INIT_RTP;}
```

After the TPR calls this special C function, it can call one or more C functions. These additional C functions can (optionally) contain the <tds.h> include file, but they cannot begin with the H_INIT_RTP macro. Figure 4-1, below, shows how the COBOL TPR calls the special C function "INIT_RTP" before the other C functions.



```
CALL "INIT_RTP".
    ...

CALL "F1" USING ...
    ...

CALL "F2" USING ...
```

```
# include <tds.h>
void INIT_RTP ()
{H_INIT_RTP ; }
```

```
[# include <tds.h>]
void F1 (...)
   ... { ... }
```

```
[# include <tds.h>]
void F2 (...)
   ... { ... }
```

Figure 4-1. The 'INIT_RTP' Call in a COBOL TPR

5. Miscellaneous

5.1 Get or Edit a Return Code (H_CBL_UGETG4, H_STD_UEDTG4)

The FILE STATUS facility is part of ANS standard COBOL and is described in the *COBOL 85 Reference Manual*. The information provided in the FILE STATUS data item is normally sufficient to diagnose most I/O errors. However, the full return code generated by Data Management can be obtained and analyzed by the COBOL program. This is done by calling the procedure H_CBL_UGETG4 in the COBOL run-time package. This facility is not part of the ANS standard and for this reason should be avoided whenever the use of the FILE STATUS phrase provides sufficient information. The return code can be translated in symbolic form by calling the procedure "H_STD_UEDTG4". The following example shows the use of H_CBL_UGETG4 and H_STD_UEDTG4:

```
WORKING-STORAGE SECTION.
77  RET-CODE-1    USAGE COMP-1.
77  RET-CODE-2    USAGE COMP-1.
77  SYMBOLIC-RET  PIC X(30).
77  NOISE COMP-2  VALUE -1.
...
PROCEDURE DIVISION.
DECLARATIVES.
FILEA-ERROR SECTION.
    USE AFTER ERROR PROCEDURE ON FILEA.
P1.
    CALL "H_CBL_UGETG4" USING RET-CODE-1 RET-CODE-2
    IF RET-CODE-2 NOT = ZERO
       CALL "H_STD_UEDTG4" USING SYMBOLIC-RET NOISE
       DISPLAY SYMBOLIC-RET.
EX-IT.
    EXIT.
END DECLARATIVES.
MAIN SECTION.
DEBUT.
    OPEN INPUT FILEA.
...
```

The return code is a hexadecimal value. The significance of each return code value is given in the *Messages and Return Codes Directory*.
The symbolic value of a return code is of the form:

```
"RC=4B820909->DSASG 2,IFNNASG"
```

5.2 Get the Program Name (H_CBL_UGETPN)

This can be used in the Procedure Division of an externally compiled program to return the program's name (as specified in the PROGRAM-ID paragraph).

To do this, use the following statement:

```
CALL "H_CBL_UGETPN" USING data-name-1
```

There must be only one parameter. This parameter (data-name-1) must be an alphanumeric data item of 30 characters. Data-name-1 is set to the name of the program.

EXAMPLE:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. THIS-PROGRAM-NAME.
...
DATA DIVISION.
...
01 CURRENT-PROG-NAME PIC X(30).
...
PROCEDURE DIVISION.
...
    CALL "H_CBL_UGETPN" USING CURRENT-PROG-NAME.
...
```

The result of the above CALL statement is exactly the same as if one had entered the following:

```
MOVE "THIS-PROGRAM-NAME" TO CURRENT-PROG-NAME
```

NOTE:

H_CBL_UGETPN may be called from a contained program. In this case, the name returned is that of the externally compiled container program, not that of the contained program. See the example below:

❑

EXAMPLE:

```
IDENTIFICATION DIVISION.            Outermost Container Program (ABC)
PROGRAM-ID. ABC.
...
CALL "XYZ"
...
IDENTIFICATION DIVISION.                      Contained Program (XYZ)
PROGRAM-ID. XYZ.
...
CALL "H_CBL_UGETPN" USING CURRENT-PROG-NAME.
...
END PROGRAM XYZ.
...
END PROGRAM ABC.
```

The result in `CURRENT-PROG-NAME` is "`ABC`" not "`XYZ`".

❑

5.3 Get the System Identification (H_CF_USYS)

Function

H_CF_USYS gives the user the system identification information about its technical state and the physical memory size.

Format

```
CALL "H_CF_USYS"  USING  CPU-NUMBER      PIC X(8)
                  MEMORY-SIZE            COMP-2
                  HOST-NAME              PIC X(4)
                  RELEASE-NAME           PIC X(4)
                  FIRMWARE-STATUS        PIC X(6)
                  TECHNICAL-STATUS            01  STATUS
                                              02 SYS PIC X(4)
                                              02 SM  PIC X(4)
                                              02 LM  PIC X(4)
```

All the parameters must be present.

A second entry point allows only the two first parameters:

```
CALL "I_CF_USYS"  USING  CPU-NUMBER      PIC X(8)
                         MEMORY-SIZE     COMP-2
```

Description of Parameters

CPU-NUMBER	specifies an output variable to which is returned the system identification (CPU number).
MEMORY-SIZE	gives the memory size in K bytes.
HOST-NAME	specifies the site name.
RELEASE-NAME	specifies the identification of the software release.
FIRMWARE-STATUS	specifies the firmware status.
TECNICAL-STATUS	specifies an output structure to which is returned the software technical status.

Return Codes

Normal	{I}
DONE	{H}_CF_USYS successfully completed.

Abnormal
none

Comments

Except MEMORY-SIZE, all this information is on the SYSOUT banners (second to last line). This primitive may be used by any process of a multi-process step.

❑

6. TDS

For details, refer *TDS COBOL Programmer's Guide*.

6.1 Debugging Using TDS Batch Interface Procedures

TDS Batch Interface procedures provide a debugging tool which allows a batch program to simulate a terminal. The simulated terminal is known as a batch pseudo-terminal. Such a pseudo-terminal can use the protocol associated with a real terminal. The batch interface allows TDS to be debugged, without the constraints that would be imposed by using live terminals.

This allows FORMS to be used through the batch interface, provided that FORMS supports the simulated terminal.

These procedures allow communication to be established with a remote TDS.

Several pseudo-terminals can connect to TDS simultaneously. Each pseudo-terminal can be either conversational or receive-only, according to the CONVERSATION-KEY parameter passed when the batch program simulates log-on.

The interface between the batch program and TDS includes three subroutines which are called by the batch program:

- H_TP7_UBCNCT which logs on the pseudo-terminal.
- H_TP7_UBDIALOG which sends and receives messages.
- H_TP7_UBRESUME which informs TDS that the last message has been processed and that another can be received.

As from TS 7254, Batch Interface subroutine names can be written "H_TP7_xx" instead of "H_MT_xx". The old syntax "H_MT_xx" is still supported.

The data structure retrieved by COPY BATCH-MSG-AREA serves to exchange data between the program and TDS.

```
01  BATCH-MSG-AREA
    02  MESSAGE-LENGTH                         COMP-1
    02  CONVERSATION-KEY                       PIC 9.
        88  CONVERSATION                       VALUE 1
        88  NO-CONVERSATION                    VALUE 0.
        88  END-OF-SESSION                     VALUE 5.
    02  END-KEY                                PIC 9.
        88  INTERMEDIATE                       VALUE 0.
        88  END-OF-TPR                         VALUE 1.
        88  END-OF-COMMIT                      VALUE 2.
        88  END-OF-TX                          VALUE 3.
    02  ERROR-KEY                              PIC 99.
        88  NO-ERROR                           VALUE 0.
        88  ABORTED                            VALUE 1.
        88  UNKNOWN-TX                         VALUE 2.
        88  LOCAL-RESTART                      VALUE 3.
        88  TDS-RESTART                        VALUE 4.
        88  TIMEOUT                            VALUE 96.
        88  ARGERR                             VALUE 97.
        88  NOSEG                              VALUE 98.
        88  DENIED                             VALUE 99.
    02  MESSAGE-TEXT                           PIC X(1024).
    02  CONNECT-TEXT REDEFINES MESSAGE-TEXT.
        03  TDS-NAME                           PIC X(4).
        03  BATCH-NAME                         PIC X(4).
        03  PASSWORD                           PIC X(8).
        03  USER-NAME                          PIC X(8).
        03  PROJECT-NAME                       PIC X(8).
        03  ACCOUNT-NAME                       PIC X(8).
        03  TERMINAL-TYPE                      PIC X(8).
        03  NODE-NAME                          PIC X(4).
        03  FILLER                             PIC X(n).
```

Usage

- The MESSAGE-LENGTH data item identifies the length of information found in data item MESSAGE-TEXT. It must be initialized with the maximum length of MESSAGE-TEXT before the connect function (CALL "H_TP7_UBCNCT"...) is called, or with the current message length in MESSAGE-TEXT before the dialog function (CALL "H_TP7_UBDIALOG") is called.

The contents of the data item MESSAGE-LENGTH are set by the standard Batch Interface as a result of a call to the dialog ("H_TP7_UBDIALOG"), or resume function ("H_TP7_UBRESUME"). MESSAGE-LENGTH indicates the current length of the message stored in MESSAGE-TEXT by TDS. MESSAGE-LENGTH can be an input or output parameter.

- The CONVERSATION-KEY data item indicates whether the message generated by TDS gives the turn to the batch program. Before calling the connect function, the batch program must initialize this data item to CONVERSATION if the program is to operate conversationally; alternatively, if the program is to operate in a receive-only mode, CONVERSATION-KEY must be set to NO-CONVERSATION. It is an output parameter.
- The END-KEY data item is set by TDS as a result of the execution of a dialog or resume function, in order to indicate which entity (TPR, commitment unit, or transaction) is closed by the message. It is an output parameter.
- The ERROR-KEY data item is set by TDS as a result of the execution of a connect, dialog or resume function. It indicates if any error has occurred, and the type of error. It is an output parameter.

  Value 96 (TIMEOUT), is returned when an EXT-TIMEOUT value not null has been used in the second parameter structure and this delay is exhausted.

  Value 97 (ARGERR), is returned by H_TP7_UBCNCT when the second parameter is used and EXT-version is out of allowed values or if EXT-TIMEOUT is negative.

  Value 98 (NOSEG), when returned by H_TP7_UBCNCT, means that the working segment cannot be created. When returned by H_TP7_UBDIALOG or H_TP7_RESUME, it means that the working segment has not been created.
- The MESSAGE-TEXT data item contains either an input text provided by the batch program or a message sent by TDS. The input text is either an input message for a transaction or a conversational input for the connect function. This standard description is named CONNECT-TEXT. It can be an input or output parameter.
- The MESSAGE-TEXT data item length supplied by the COPY statement ranges from 52 characters to the value specified in the MESSAGE-LENGTH clause at TDSGEN. The default value is 1024 characters.
- Before the execution of the connect function, the data structure named CONNECT-TEXT is to be set as follows:

  TDS-NAME should contain the name of the TDS subsystem to be accessed.

  BATCH-NAME should be a unique system name identifying the pseudo-terminal as a correspondent of the communication facility. This name does not need to be described in the network generation; it has the same format as a terminal name.

PASSWORD should contain the password associated with the user name specified in the USER-NAME field.

USER-NAME should contain the unique name of a user known by the specified TDS subsystem; this user name is used in a similar way as for a terminal user.

PROJECT-NAME and ACCOUNT-NAME should contain the project and billing under which the user intends to work. When default project and/or billing are to be used, the corresponding data item should contain spaces.

TERMINAL-TYPE, when requested to simulate a true terminal, may contain the name of a supported terminal type. The following terminals are supported: DKU7007, DKU7107, DKU7211, IBM3270, IBM3278/3279, VIP7804, PC7800, and Minitel.

If the terminal type specified is erroneous, the connect function is denied and an error-key DENIED is returned.

If this field is completed, the functions and protocols relevant to terminals apply to the simulated terminal.

- The user must specify specific terminal character strings, such as, message headers and device protocol headers (STA, FC1, FC2).
- FORMS is accessible to simulated terminals if the terminal type is supported by the current version of FORMS. The program will receive data in the specific format of the terminal.
- NODE-NAME is an optional field but if present it must contain a valid node name as generated in the Network. This name is the node name under which the TDS application is running and enables a batch application to be connected to a remote TDS.
- FILLER: (n) = 976 or 972 if node name is specified.

6.2 CONNECT Function (H_TP7_UBCNCT)

Syntax

```
CALL "H_TP7_UBCNCT" USING BATCH-MSG-AREA [ EXT-AREA ].
```

Description

The connect function must be performed as the first batch-entry function and is similar to a terminal user log-on.

Usage

- The connect function may be performed only once and must be successfully executed before the dialog and resume functions.
- The BATCH-MSG-AREA data structure is used in a specific manner prior to execution of the connect function.

  The MESSAGE-LENGTH field contains the maximum length of the MESSAGE-TEXT data item.

  The CONVERSATION-KEY field specifies the mode (receive-only or conversational) in which the program will operate.

  The MESSAGE-TEXT field contains the data structure identified as CONNECT-TEXT.
- Upon completion of the connect function, the ERROR-KEY data item should be checked for successful completion (NO-ERROR). After an unsuccessful connect no other functions may be performed.
- The CONVERSATION-KEY data item will indicate if the next function should be dialog or resume upon completion of the connect function.

- The EXT-AREA is used to pass a 12-character password, or 12-character username, and to define a timeout value and a status area. It has the following format:

```
   01  EXT-AREA.
       02  EXT-VERSION                COMP-1.
       02  EXT-PASSWORD               PIC X(12).
       02  EXT-TIMEOUT                COMP-1.
       02  EXT-USERNAME               PIC X(12).
       02  EXT-PROJECTNAME            PIC X(12).
       02  EXT-ACCOUNTNAME            PIC X(12).
       02  EXT-STATUS-AREA.
           03 EXT-ERRORID             PIC X(2).
           03 EXT-RETURN-CODE         PIC X(30).
           03 EXT-STAT                PIC X(2).
           03 EXT-REASON              PIC X(4).
           03 EXT-COMP-STAT           PIC X(2).
```

If EXT-AREA is absent, an 8-character password and an 8-character username are taken from the BATCH-MSG-AREA.

If EXT-VERSION is 1, only the EXT-PASSWORD field is taken into account (the password is taken from this field).

If EXT-VERSION is 2, both the EXT-PASSWORD and EXT-TIMEOUT fields are used. The connection will be done with a TIMEOUT value (if a positive value is supplied). This timeout value, specified in seconds (to connection time), is supplied in EXT-TIMEOUT. If EXT-TIMEOUT is null (0), no timer detection upon connection is done. The value is limited by the COMP-1 capacity (about 9 hours). The EXT-PASSWORD must be filled with 12 characters.

If EXT-VERSION is 3, the fields EXT-PASSWORD, EXT-TIMEOUT, EXT-USERNAME, EXT-PROJECTNAME, and EXT-ACCOUNTNAME of this extended area are taken into account and must be filled. If no timeout detection is desired, EXT-TIMEOUT must be set to 0 (zero). If default project and/or billing are to be used, then EXT-PROJECTNAME and EXT-ACCOUNTNAME may be set to blanks.

If EXT-VERSION is 4, the fields of EXT-STATUS-AREA are available. EXT-STATUS-AREA in an output area used to help you debug problems when finalizing a Batch Interface program.

If EXT-VERSION is not 1, 2, 3, or 4 or if EXT-TIMEOUT is negative or an exception has been detected on the parameters provided by the caller, the ERROR-KEY is filled with ARGERR (97).

If EXT-TIMEOUT is not null (0) and no response to the connection received passed this delay, ERROR-KEY is filled with TIME-OUT (96).

If the EXT-AREA structure is to be used with the "H_TP7_UBDIALOG" subroutine, the DVCHD-AREA structure must be filled (if no device header is to be sent, STRUCT-LGT should be set to 0).

The fields EXT-PASSWORD and EXT-ACCOUNTNAME are used (in input) by the "H_TP7_UBCNCT" routine only. These fields are neither taken into account nor modified by the subroutines "H_TP7_UBDIALOG" and "H_TP7_UBRESUME".

EXT-ERRORID is a unique identifier for each abnormal return from the Batch Interface subroutine.

EXT-RETURN-CODE is the last return code received (in case of anomaly it is shown in edited format). Its value may be DONE.

EXT-STAT is the status of the VCAM verb returning the error state.

EXT-REASON is the reason for the disconnection or the connection rejection.

The contents of EXT-COMP-STAT depends on the value of EXT-ERRORID. It is explained below.

EXT-ERRORID Values Returned by "H_TP7_UBCNCT"

"00"	No error. All fields of EXT-STATUS-AREA are loaded with the character "0".
"01"	Problem when activating the workstation. The following status fields are meaningful: EXT-RETURN-CODE (CHECK,ARGERR) When "CHECK", see the EXT-STAT value. When "ARGERR", contact your Service Center. **EXT-STAT Values:** "32"X: The workstation is already active (name is already known). Action: Verify name and/or check program logic. "33"X: The workstation is de-activating. Action: Contact your Service Center. "34"X: System resource overload. Action: Contact your Service Center.
"02"	Problem with model of declared terminal. The following status fields are meaningful: EXT-RETURN-CODE (NOMATCH, other RC) When "NOMATCH", the terminal type is unknown. Action: Check the terminal type value with the H_TERM subfile of SYS.HSLLIB. When "other RC", contact your Service Center.

"03"	Problem when activating mailbox. The following status fields are meaningful: EXT-RETURN-CODE (CHECK, ARGERR) When "CHECK", see the EXT-STAT value. When "ARGERR", contact your Service Center. **EXT-STAT Values:** "32"X: The mailbox is already active (name is already known). Action: Verify name and/or check program logic. "33"X: The mailbox is de-activating. Action: Contact your Service Center. "34"X: System resource overload. Action: Contact your Service Center.
"04"	Problem when opening message group. The following status fields are meaningful: EXT-RETURN-CODE (CHECK, ARGERR) When "CHECK", see the EXT-STAT value. When "ARGERR", contact your Service Center. **EXT-STAT Values:** "34"X: System resource overload. Action: Contact your Service Center. "35"X: Connection reject. Action: See the reason for the connection rejection in EXT-REASON.
"05"	Problem on reception of event during connection phase. The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"06"	OPENACK event was expected, but another one was received. The following status fields are meaningful: EXT-RETURN-CODE, EXT-COMP-STAT (received event) Action: Contact your Service Center.

"07"	OPENACK REJECT (connection is refused in the second phase). The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT (reject code "35"X), EXT-REASON Action: See the reason for the rejection in EXT-REASON.
"08"	Problem at connection negotiation ($H_INQUIR VCAM verb) The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.

EXT-ERRORID Values Returned by Other Subroutines

"09"	Erroneous subroutine call in this context. The subroutine has probably been called while the Batch Interface was already in error. No status fields are meaningful. Action: Check your program.
"10"	BATCH-MSG-AREA area is not the same as the one used at the last call to the "H_TP7_UBCNCT" subroutine (control is done on address). No status fields are meaningful. Action: Check your program.
"11"	H_TP7_UBDIALOG subroutine has been called instead of H_TP7_UBRESUME (conversation key was "NO-CONVERSATION"). No status fields are meaningful. Action: Check your program.
"12"	Input message length is negative. No status fields are meaningful. Action: Check your program.

"13"	H_TP7_UBRESUME subroutine has been called instead of H_TP7_UBDIALOG (conversation key was "CONVERSATION"). No status fields are meaningful. Action: Check your program.
"14"	Abnormal status when receiving an interruption during another VCAM primitive. The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"15"	Abnormal status (neither done nor interrupt pending) when sending or receiving a message. The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"16"	Abnormal status (neither v_done nor v_skip) when receiving an event. The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"17"	Disconnection occurred (V_MSGCLOSED has been received). The following status fields are meaningful: EXT-RETURN-CODE, EXT-REASON Action: None, mailbox and workstation have been de-activated.
"18"	Abnormal status when receiving an interruption after a V_INTERRUPT event. The following status fields are meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"19"	Unexpected event received by the Batch Interface. The following status field is meaningful: EXT-COMP-STAT (received event) Action: Contact your Service Center.

"20"	Status neither done nor moredata when receiving a message. The following status field is meaningful: EXT-RETURN-CODE, EXT-STAT Action: Contact your Service Center.
"21"	Detected level not supported. The following status field is meaningful: EXT-COMP-STAT (detected level) Action: Contact your Service Center.

Description of EXT-REASON Possible Values

"01"X	Abnormal rejection.
"02"X	Destination node not operable.
"03"X	Destination node saturated.
"04"X	Mailbox unknown.
"05"X	Mailbox not operable.
"06"X	Mailbox saturated.
"07"X	Destination application saturated.
"09"X	Dialog rejection (as a result of negotiation).
"0A"X	Presentation rejection (as a result of negotiation).
"15"X	Timeout.
"18"X	Security violation.
"40"X	Destination node unknown.
"41"X	Path to the destination node is not available.
"42"X	Duplicate user identifier.

6.3 DIALOG Function (H_TP7_UBDIALOG)

6.3.1 DIALOG Function Without the Device Header

Syntax

```
CALL "H_TP7_UBDIALOG" USING BATCH-MSG-AREA, [DVCHD-AREA EXT-AREA].
```

Description

Sends a message to TDS and awaits the corresponding reply.

Usage

- The dialog function may be executed only when the previous statement (Call H_TP7_UBCNCT, or Call H_TP7_UBDIALOG) has obtained a CONVERSATION-KEY set to CONVERSATION.
- Before the function is executed, the message to be sent to TDS must be moved to the MESSAGE-TEXT field and MESSAGE-LENGTH be set to the appropriate value. After the function is executed, the reply from TDS is available in the MESSAGE-TEXT field, the length of which is stored in MESSAGE-LENGTH.
- If the EXT-AREA structure is to be used, the DVCHD-AREA must be filled in (STRUCT-LGT set to 0).

6.3.2 DIALOG Function With the Device Header

Syntax

```
CALL "H_TP7_UBDIALOG" USING BATCH-MSG-AREA DVCHD-AREA, [EXT-AREA].
```

Description

Sends a message to TDS with DEVICE HEADER and awaits the corresponding reply.

Usage

In addition to the DIALOG function just described, the following structure must be filled:

```
01 DVCHD-AREA.
  02 STRUCT-LGT COMP-1.
  02 HEADER.
  03 HEADER-LGT COMP-1.
  03 HEADER-VL PIC X (i).
```

- The STRUCT-LGT data item defines the length of the structure and must be equal to HEADER-LGT + 4.
- The HEADER-LGT data item defines the length of the value in HEADER-VL and is always even because two characters give one hexadecimal byte.
- HEADER-VL contains the device header value whose PIC must not exceed 30. The values specified in the picture string must be a hexadecimal value.

EXAMPLE:

```
MOVE 6 TO HEADER-LGT.
MOVE "7DD7C7" TO HEADER-VL.
MOVE 10 TO STRUCT-LGT
```

❑

Remarks

The device header is not tested for terminal type and the terminal is forced to operate in unedited mode. Thus, in line mode, the device header is added before the input text and must be taken into account by TPRs.

6.4 RESUME Function (H_TP7_UBRESUME)

Syntax

```
CALL "H_TP7_UBRESUME" USING BATCH-MSG-AREA, [EXT-AREA].
```

Description

Notifies TDS that the message received is processed and that the BATCH-MSG-AREA is available for a new message.

Usage

- The resume function may be executed only if CONVERSATION-KEY is set to NO-CONVERSATION upon the return of H_TP7_UBCNCT, or H_TP7_UBDIALOG, or H_TP7_UBRESUME.
- After the execution of the function, the message from TDS is available in the MESSAGE-TEXT field, the length of which is stored in MESSAGE-LENGTH.

7. Administrative Utilities Programmatic Interface (AUPI)

For a detailed description of how to use AUPI procedures, refer to the *AUPI User's Guide*.

This chapter deals with the following procedure COBOL calls:

H_NA_ICLCR	Close-Correspondent
H_NA_ICREFL	Create-Filter
H_NA_IDEFL	Delete-Filter
H_NA_IGETFD	Receive-Field
H_NA_IGETHD	Receive-Header
H_NA_IGETINF	Get-System-Information
H_NA_IGETPOS	Set-Position
H_NA_IGETREC	Receive-Record
H_NA_IGETVB	Receive-Verbatim
H_NA_IOPCR	Open-Correspondent
H_NA_ISENDFD	Send-Field
H_NA_ISENDHD	Send-Header
H_NA_ISENVB	Send-Verbatim

7.1 Data Structures COBOL Descriptions

7.1.1 AF-AUPI-FIELD

Function

To declare the AUPI field used by H_NA_IGETFD and H_NA_ISENDFD.

Format

```
COPY AF-AUPI-FIELD.
```

Expansion of the Primitive: COPY AF-AUPI-FIELD

```
01 AF-AUPI-FIELD.
  02 AF-REVISION            COMP-1          VALUE 0.
  02 AF-NEXT-FLAG           PIC X           VALUE "N".
      88 AF-ADDR                            VALUE "A".
      88 AF-NEXT                            VALUE "N".
      88 AF-LABEL                           VALUE "L".
  02 AF-RIF.
    03 AF-REGION            PIC X           VALUE "R".
      88 AF-SELECTION                       VALUE "S".
      88 AF-MODIFICATION                    VALUE "M".
      88 AF-RESPONSE                        VALUE "R".
      88 AF-ERROR                           VALUE "E".
    03 AF-ITEM              COMP-1          VALUE 0.
    03 AF-FIELD             COMP-1          VALUE 0.
    03 AF-LABEL             PIC X(8)        VALUE " ".
  02 AF-NUMERIC-LENGTH      COMP-1          VALUE 0.
  02 AF-CHAR-LENGTH         COMP-1          VALUE 0.
  02 AF-BASIC-TYPE          PIC X           VALUE " ".
      88 AF-CHARACTER-TYPE                  VALUE "C".
      88 AF-NUMERIC-TYPE                    VALUE "N".
      88 AF-OBJECT-TYPE                     VALUE "O".
      88 AF-BOTH-TYPE                       VALUE "B".
  02 AF-FILLER-0            PIC X           VALUE " ".
  02 AF-NUMERIC-FIELD       COMP-2          VALUE 0.
  02 AF-CHAR-FIELD          LX(255)  DEPENDING ON AF-CHAR-LENGTH.
  02 AF-DESC-TYPE           COMP-1          VALUE 0.
  02 AF-STRING-SUBTYPE      PIC X           VALUE " ".
  02 AF-FILLER-1            PIC X           VALUE " ".
```

7.1.2 AF-AUPI-FILTER

Function

To declare the AUPI filter data structure used by H_NA_ICREFL and H_NA_IDELFL. The AUPI filter is associated with an Administrative Correspondent.

Format

```
COPY AF-AUPI-FILTER.
```

Expansion of the Primitive: COPY AF-AUPI-FILTER

```
01     AF-AUPI-FILTER.
  02     AF-REVISION            COMP-1           VALUE 0.
  02     AF-FL-NAME             PIC X(8)         VALUE " ".
  02     AF-FL-TYPE             PIC X(2)         VALUE "OU".
  02     AF-FL-LOGIC            PIC X(4)         VALUE "INCL".
      88   AF-INCLUSIVE                          VALUE "INCL".
      88   AF-EXCLUSIVE                          VALUE "EXCL".
      88   AF-OBLIGATORY                         VALUE "OBLI".
  02     AF-DOMAIN.
    03     AF-DOMAIN-L          COMP-1           VALUE -1.
    03     AF-DOMAIN-H          COMP-1           VALUE -1.
  02     AF-POWER.
    03     AF-POWER-L           COMP-1           VALUE -1.
    03     AF-POWER-H           COMP-1           VALUE -1.
  02     AF-CLASS.
    03     AF-CLASS-L           COMP-1           VALUE -1.
    03     AF-CLASS-H           COMP-1           VALUE -1.
  02     AF-CODE.
    03     AF-CODE-L            COMP-1           VALUE -1.
    03     AF-CODE-H            COMP-1           VALUE -1.
  02     AF-LEVEL.
    03     AF-LEVEL-L           COMP-1           VALUE -1.
    03     AF-LEVEL-H           COMP-1           VALUE -1.
  02     AF-SYSTEM.
    03     AF-SYSTEM-1          PIC X(4)         VALUE " ".
    03     AF-SYSTEM-2          PIC X(4)         VALUE " ".
  02     AF-NAME.
    03    AF-NAME-1            PIC X(8)         VALUE " ".
    03    AF-NAME-2            PIC X(8)         VALUE " ".
  02    AF-VALUE.
    03    AF-VALUE-1           COMP-1           VALUE -1.
    03    AF-VALUE-2           COMP-1           VALUE -1.
  02    AF-TIME.
    03    AF-TIME-1            COMP-2           VALUE 0.
    03    AF-TIME-2            COMP-2           VALUE 0.
```

NOTE:

Although the field AF-FL-LOGIC in the COPY AF-AUPI-FILTER expansion can take the keyword value OBLI (obligatory), the equivalent NETGEN declaration is LOGIC=MAND (mandatory) in the FL directive.

7.1.3 AH-AUPI-HEADER

Function

To declare the AUPI header data structure used by H_NA_IGETHD and H_NA_ISENDHD.

Format

```
COPY AH-AUPI-HEADER.
```

Expansion of the Primitive: COPY AH-AUPI-HEADER

```
01    AH-AUPI-HEADER.
  02    AH-REVISION                    COMP-1     VALUE 0.
  02    AH-REGION                      PIC X      VALUE "H".
      88  AH-HEADER                               VALUE "H".
  02    AH-FILLER-0                    PIC X      VALUE " ".
  02    AH-PROTOCOL-VERSION            COMP-1     VALUE 0.
      88  AH-AEP2                                 VALUE 2.
  02    AH-RESPONDER.
    03    AH-R-SYSTEM-TYPE             COMP-1     VALUE 0.
      88  AH-UNKNOWN                              VALUE  0.
      88  AH-DNS                                  VALUE  1.
      88  AH-GCOS6                                VALUE  2.
      88  AH-GCOS7                                VALUE  3.
      88  AH-GCOS8                                VALUE  5.
      88  AH-CNS                                  VALUE 10.
      88  AH-NAS400                               VALUE 11.
      88  AH-NCC                                  VALUE 12.
      88  AH-SPIX                                 VALUE 13.
      88  AH-OTHER                                VALUE 15.
    03    AH-R-SOFTWARE-LEVEL          COMP-1     VALUE 0.
    03    AH-R-SYSTEM-ID               PIC X(4)   VALUE " ".
    03    AH-R-TIME-RESPONSE           COMP-2     VALUE 0.
```

```
02    AH-ORIGIN.
  03    AH-O-SYSTEM-ID             PIC X(4)   VALUE " ".
  03    AH-O-TYPE                  COMP-1     VALUE 0.
    88  AH-UNSOLICITED-MSG                    VALUE  1.
    88  AH-NCC-MSG                            VALUE  2.
    88  AH-NOI-MSG                            VALUE  3.
    88  AH-LOG-MSG                            VALUE  4.
    88  AH-ROM-MSG                            VALUE  5.
    88  AH-RFU-MSG                            VALUE  6.
    88  AH-CMD-MSG                            VALUE  7.
    88  AH-EX-MSG                             VALUE  8.
    88  AH-ADM-MSG                            VALUE  9.
  03    AH-O-SPECIFIC              COMP-1     VALUE 0.
02    AH-COMMAND-DESCR.
  03    AH-C-RFU                   COMP-1     VALUE 0.
  03    AH-C-FUNCTION.
    04    AH-C-DOMAIN              COMP-1     VALUE 0.
    88  AH-DSA-COMMUNICATION                  VALUE  0.
    88  AH-ADM-LOC-SYSTEM                     VALUE  1.
    88  AH-APPL-ADM                           VALUE  2.
    88  AH-ADM-CTRL                           VALUE  3.
    88  AH-SECURITY-ADM                       VALUE  4.
    04    AH-C-POWER               COMP-1     VALUE 0.
  03    AH-C-CLASS                 COMP-1     VALUE 0.
    88  AH-CLASS-XC                           VALUE  1.
    88  AH-CLASS-XN                           VALUE  2.
    88  AH-CLASS-SU                           VALUE  3.
    88  AH-CLASS-PL                           VALUE  4.
    88  AH-CLASS-LL                           VALUE  5.
    88  AH-CLASS-NS                           VALUE  6.
    88  AH-CLASS-NR                           VALUE  7.
    88  AH-CLASS-VC                           VALUE  8.
    88  AH-CLASS-TS                           VALUE  9.
    88  AH-CLASS-MB                           VALUE 10.
    88  AH-CLASS-SS                           VALUE 11.
    88  AH-CLASS-LC                           VALUE 12.
    88  AH-CLASS-CT                           VALUE 13.
    88  AH-CLASS-DV                           VALUE 14.
    88  AH-CLASS-AF                           VALUE 15.
    88  AH-CLASS-FX                           VALUE 18.
    88  AH-CLASS-EX                           VALUE 19.
    88  AH-CLASS-RS                           VALUE 19.
    88  AH-CLASS-SY                           VALUE 20.
    88  AH-CLASS-TC                           VALUE 21.
    88  AH-CLASS-WS                           VALUE 22.
    88  AH-CLASS-NC                           VALUE 23.
    88  AH-CLASS-SD                           VALUE 24.
    88  AH-CLASS-CH                           VALUE 25.
    88  AH-CLASS-SN                           VALUE 26.
    88  AH-CLASS-CL                           VALUE 27.
    88  AH-CLASS-LK                           VALUE 28.
```

```
88  AH-CLASS-PC                             VALUE 29.
88  AH-CLASS-SC                             VALUE 30.
88  AH-CLASS-CC                             VALUE 31.
88  AH-CLASS-FL                             VALUE 34.
88  AH-CLASS-SB                             VALUE 35.
88  AH-CLASS-NA                             VALUE 36.
88  AH-CLASS-MU                             VALUE 38.
88  AH-CLASS-AC                             VALUE 43.
88  AH-CLASS-SR                             VALUE 44.
88  AH-CLASS-LD                             VALUE 45.
88  AH-CLASS-OP                             VALUE 48.
88  AH-CLASS-LG                             VALUE 49.
88  AH-CLASS-TL                             VALUE 51.
88  AH-CLASS-DF                             VALUE 53.
88  AH-CLASS-FT                             VALUE 54.
88  AH-CLASS-AG                             VALUE 56.
88  AH-CLASS-CO                             VALUE 57.
88  AH-CLASS-PS                             VALUE 58.
88  AH-CLASS-SG                             VALUE 59.
88  AH-CLASS-SX                             VALUE 60.
88  AH-CLASS-LX                             VALUE 61.
88  AH-CLASS-DX                             VALUE 62.
88  AH-CLASS-UD                             VALUE 63.
88  AH-CLASS-LN                             VALUE 64.
88  AH-CLASS-CD                             VALUE 65.
88  AH-CLASS-TX                             VALUE 66.
88  AH-CLASS-DP                             VALUE 68.
88  AH-CLASS-WM                             VALUE 69.
88  AH-CLASS-TU                             VALUE 70.
88  AH-CLASS-MD                             VALUE 71.
88  AH-CLASS-SW                             VALUE 74.
88  AH-CLASS-NU                             VALUE 75.
88  AH-CLASS-AD                             VALUE 86.
88  AH-CLASS-ET                             VALUE 87.
88  AH-CLASS-SP                             VALUE 88.
88  AH-CLASS-CB                             VALUE 90.
88  AH-CLASS-AI                             VALUE 91.
88  AH-CLASS-AL                             VALUE 95.
88  AH-CLASS-AP                             VALUE 96.
88  AH-CLASS-AX                             VALUE 97.
88  AH-CLASS-UT                             VALUE 98.
88  AH-CLASS-VH                             VALUE 100.
88  AH-CLASS-UA                             VALUE 101.
88  AH-CLASS-AS                             VALUE 102.
88  AH-CLASS-IS                             VALUE 105.
88  AH-CLASS-ID                             VALUE 106.
88  AH-CLASS-RQ                             VALUE 110.
88  AH-CLASS-RB                             VALUE 111.
88  AH-CLASS-ML                             VALUE 112.
88  AH-CLASS-MK                             VALUE 113.
88  AH-CLASS-IK                             VALUE 114.
```

```
            88  AH-CLASS-NK                            VALUE 115.
            88  AH-CLASS-FR                            VALUE 120.
            88  AH-CLASS-FS                            VALUE 121.
            88  AH-CLASS-FE                            VALUE 122.
            88  AH-CLASS-FM                            VALUE 123.
            88  AH-CLASS-PA                            VALUE 130.
            88  AH-CLASS-MA                            VALUE 140.
            88  AH-CLASS-MQ                            VALUE 141.
            88  AH-CLASS-MM                            VALUE 142.
            88  AH-CLASS-MI                            VALUE 143.
            88  AH-CLASS-FA                            VALUE 150.
            88  AH-CLASS-FF                            VALUE 151.
            88  AH-CLASS-FD                            VALUE 152.
            88  AH-CLASS-DA                            VALUE 160.
            88  AH-CLASS-SA                            VALUE 190.
            88  AH-CLASS-SV                            VALUE 191.
            88  AH-CLASS-QD                            VALUE 192.

      ***************************************************************
      *                                                             *
      *    THIS PART ENSURES COMPATIBILITY WITH PREVIOUS RELEASES   *
      *                                                             *
      ***************************************************************
            88  AH-PL                                  VALUE  4.
            88  AH-LL                                  VALUE  5.
            88  AH-NS                                  VALUE  6.
            88  AH-NR                                  VALUE  7.
            88  AH-VC                                  VALUE  8.
            88  AH-TS                                  VALUE  9.
            88  AH-MB                                  VALUE 10.
            88  AH-SS                                  VALUE 11.
            88  AH-LC                                  VALUE 12.
            88  AH-CT                                  VALUE 13.
            88  AH-DV                                  VALUE 14.
            88  AH-AF                                  VALUE 15.
            88  AH-EX                                  VALUE 19.
            88  AH-SY                                  VALUE 20.
            88  AH-TC                                  VALUE 21.
            88  AH-MO                                  VALUE 22.
            88  AH-WS                                  VALUE 22.
            88  AH-NC                                  VALUE 23.
            88  AH-SD                                  VALUE 24.
            88  AH-CH                                  VALUE 25.
            88  AH-SN                                  VALUE 26.
            88  AH-CL                                  VALUE 27.
            88  AH-LK                                  VALUE 28.
            88  AH-PC                                  VALUE 29.
            88  AH-SC                                  VALUE 30.
            88  AH-CC                                  VALUE 31.
            88  AH-FL                                  VALUE 34.
```

```
             88  AH-SB                                 VALUE 35.
             88  AH-MU                                 VALUE 38.
             88  AH-PP                                 VALUE 40.
             88  AH-AC                                 VALUE 43.
             88  AH-SR                                 VALUE 44.
             88  AH-LD                                 VALUE 45.
             88  AH-FM                                 VALUE 46.
             88  AH-OP                                 VALUE 48.
             88  AH-LG                                 VALUE 49.
             88  AH-TL                                 VALUE 51.
             88  AH-FT                                 VALUE 54.
             88  AH-AG                                 VALUE 56.
             88  AH-CO                                 VALUE 57.
             88  AH-PS                                 VALUE 58.
             88  AH-SX                                 VALUE 60.
             88  AH-LX                                 VALUE 61.
             88  AH-DX                                 VALUE 62.
             88  AH-UD                                 VALUE 63.
             88  AH-LN                                 VALUE 64.
             88  AH-CD                                 VALUE 65.
             88  AH-TX                                 VALUE 66.
             88  AH-DP                                 VALUE 68.
             88  AH-WM                                 VALUE 69.
             88  AH-TU                                 VALUE 70.
             88  AH-MD                                 VALUE 71.
             88  AH-SW                                 VALUE 74.
             88  AH-NU                                 VALUE 75.
             88  AH-CB                                 VALUE 90.
             88  AH-AL                                 VALUE 95.
             88  AH-UT                                 VALUE 98.
             88  AH-IS                                 VALUE 105.
             88  AH-ID                                 VALUE 106.
             88  AH-RB                                 VALUE 111.
             88  AH-ML                                 VALUE 112.
             88  AH-MK                                 VALUE 113.
             88  AH-IK                                 VALUE 114.
             88  AH-NK                                 VALUE 115.
           03    AH-C-CODE                  COMP-1     VALUE 0.
             88  AH-CODE-NB                            VALUE  1.
             88  AH-CODE-LS                            VALUE  2.
             88  AH-CODE-DA                            VALUE  3.
             88  AH-CODE-HR                            VALUE  4.
             88  AH-CODE-GH                            VALUE  4.
             88  AH-CODE-UP                            VALUE  5.
             88  AH-CODE-MP                            VALUE  6.
             88  AH-CODE-CR                            VALUE  7.
             88  AH-CODE-OP                            VALUE  7.
             88  AH-CODE-EX                            VALUE  7.
             88  AH-CODE-DL                            VALUE  8.
             88  AH-CODE-CL                            VALUE  8.
             88  AH-CODE-GA                            VALUE 12.
```

```
            88  AH-CODE-ER                              VALUE 14.
            88  AH-CODE-TH                              VALUE 15.
            88  AH-CODE-OF                              VALUE 17.
            88  AH-CODE-TX                              VALUE 50.
            88  AH-CODE-SU                              VALUE 50.
            88  AH-CODE-SF                              VALUE 51.
            88  AH-CODE-RL                              VALUE 51.
            88  AH-CODE-DS                              VALUE 51.
            88  AH-CODE-SW                              VALUE 51.
            88  AH-CODE-DE                              VALUE 52.
            88  AH-CODE-LD                              VALUE 52.
            88  AH-CODE-AS                              VALUE 52.
            88  AH-CODE-RI                              VALUE 52.
            88  AH-CODE-DF                              VALUE 53.
            88  AH-CODE-DP                              VALUE 53.
            88  AH-CODE-RE                              VALUE 54.
            88  AH-CODE-ST                              VALUE 54.
            88  AH-CODE-TR                              VALUE 55.
            88  AH-CODE-TF                              VALUE 56.
            88  AH-CODE-DI                              VALUE 57.
            88  AH-CODE-CS                              VALUE 60.
            88  AH-CODE-FC                              VALUE 100.
            88  AH-CODE-FD                              VALUE 101.
            88  AH-CODE-FN                              VALUE 102.
            88  AH-CODE-AC                              VALUE 103.
            88  AH-CODE-EM                              VALUE 150.
            88  AH-CODE-ED                              VALUE 151.
            88  AH-CODE-DR                              VALUE 160.

      *****************************************************************
      *    THIS PART ENSURES COMPATIBILITY WITH PREVIOUS RELEASES    *
      *                                                               *
      *****************************************************************
            88  AH-NB                                   VALUE  1.
            88  AH-LS                                   VALUE  2.
            88  AH-DA                                   VALUE  3.
            88  AH-GH                                   VALUE  4.
            88  AH-UP                                   VALUE  5.
            88  AH-MP                                   VALUE  6.
            88  AH-CR                                   VALUE  7.
            88  AH-DL                                   VALUE  8.
            88  AH-GA                                   VALUE 12.
          03    AH-C-LENGTH                 COMP-1      VALUE 0.
          03    AH-C-SELECTORS              COMP-1      VALUE 0.
          03    AH-C-MODIFIERS              COMP-1      VALUE 0.
```

```
  02    AH-RESPONSE-DESCR.
    03    AH-R-FORMAT.
      04    AH-R-REJECTED-BEFORE       PIC 9      VALUE 0.
      04    AH-R-REJECTED-DURING       PIC 9      VALUE 0.
      04    AH-R-FILLER-1              PIC 99     VALUE 0.
      04    AH-R-INCOMPLETE            PIC 9      VALUE 0.
      04    AH-R-MORE-COMING           PIC 9      VALUE 0.
      04    AH-R-CONTINUATION          PIC 9      VALUE 0.
      04    AH-R-FILLER-2              PIC 9      VALUE 0.
    03    AH-R-IMPORTANCE              COMP-1     VALUE 0.
    03    AH-R-LENGTH                  COMP-1     VALUE 0.
    03    AH-R-ITEMS-IN-RESPONSE       COMP-1     VALUE 0.
    03    AH-R-FIELDS-PER-ITEM         COMP-1     VALUE 0.
  02    AH-ERROR-DESCR.
    03    AH-E-LENGTH                  COMP-1     VALUE 0.
    03    AH-E-ERRORS                  COMP-1     VALUE 0.
```

7.1.4 AR-AUPI-RECORD

Function

To declare the AUPI record data structure returned to the AUT by H_NA_IGETREC.

Format

```
COPY AR-AUPI-RECORD.
```

Expansion of the Primitive: COPY AR-AUPI-RECORD

```
01    AR-AUPI-RECORD .
  02    AR-REVISION          COMP-1           VALUE 0.
  02    AR-LENGTH-RECORD     COMP-1           VALUE 0.
  02    AR-CHAR-RECORD       PIC X(200)       VALUE " ".
```

7.1.5 AS-AUPI-STATUS

Function

To declare the AUPI status data structure used by all AUPI procedures.

Format

```
COPY AS-AUPI-STATUS.
```

Expansion of the Primitive: COPY AS-AUPI-STATUS

```
01    AS-AUPI-STATUS.
  02    AS-REVISION                     COMP-1     VALUE 0.
  02    AS-RETURNED-STATUS.
    03    AS-AUPI-FUNCTION              PIC X(2)   VALUE " ".
      88  AS-FUNCTION-OPEN                         VALUE "OP".
      88  AS-FUNCTION-CLOSE                        VALUE "CL".
      88  AS-FUNCTION-RCV-HDR                      VALUE "RH".
      88  AS-FUNCTION-RCV-FIELD                    VALUE "RF".
      88  AS-FUNCTION-RCV-VBTM                     VALUE "RV".
      88  AS-FUNCTION-CR-FILTER                    VALUE "CF".
      88  AS-FUNCTION-DL-FILTER                    VALUE "DF".
      88  AS-FUNCTION-SET-POS                      VALUE "SP".
      88  AS-FUNCTION-SEND-HDR                     VALUE "SH".
      88  AS-FUNCTION-SEND-FIELD                   VALUE "SF".
      88  AS-FUNCTION-SEND-VBTM                    VALUE "SV".
      88  AS-FUNCTION-GET-INF                      VALUE "SI".
      88  AS-FUNCTION-RCV-REC                      VALUE "RR".
    03    AS-MAJOR-STATUS               COMP-1     VALUE  0.
      88  AS-OK                                    VALUE  0.
      88  AS-REGION-NOT-PRESENT                    VALUE  1.
      88  AS-ITEM-NOT-PRESENT                      VALUE  2.
      88  AS-FIELD-NOT-PRESENT                     VALUE  3.
      88  AS-END-OF-MESSAGE                        VALUE  4.
      88  AS-NEED-NCL                              VALUE  5.
      88  AS-SEQUENCE-ERROR                        VALUE  6.
      88  AS-LACK-OF-RESOURCE                      VALUE  7.
      88  AS-REQ-TIMEOUT                           VALUE  8.
      88  AS-INVALID-INPUT                         VALUE  9.
      88  AS-INVALID-DOPE-VECTOR                   VALUE 10.
      88  AS-SHORT-ARGUMENT                        VALUE 11.
      88  AS-DUPLICATE-FIELD                       VALUE 12.
      88  AS-NCL-SYNTAX-ERROR                      VALUE 13.
      88  AS-NON-AUPI-SESSION-EVENT                VALUE 14.
      88  AS-NON-AUPI-EVENT                        VALUE 15.
```

```
        88  AS-SESSION-ESTABLISHED                  VALUE 16.
        88  AS-END-OF-SESSION                       VALUE 17.
        88  AS-UNKNOWN-ACID                         VALUE 18.
        88  AS-UNKNOWN-STD                          VALUE 19.
        88  AS-UNKNOWN-AEP-VALUE                    VALUE 20.
        88  AS-OK-TO-SEND                           VALUE 21.
        88  AS-OK-TO-RECEIVE                        VALUE 22.
        88  AS-AUPI-INTERNAL-ERROR                  VALUE 23.
        88  AS-INCOMPLETE-RECORD                    VALUE 24.
        88  AS-END-OF-FILE                          VALUE 25.
        88  AS-FILE-ERROR                           VALUE 26.
        88  AS-UNKNOWN-CORRESPONDENT                VALUE 27.
        88  AS-NO-CURRENT-MSG                       VALUE 28.
        88  AS-WINDOW-OVERFLOW                      VALUE 29.
        88  AS-FILE-NOT-OPENED                      VALUE 31.
        88  AS-FILE-ALREADY-OPEN                    VALUE 32.
        88  AS-ASF-LOCKED                           VALUE 33.
        88  AS-MISSING-MANDATORY-PARAMETER          VALUE 34.
        88  AS-SCID-NOT-FOUND                       VALUE 35.
        88  AS-INVALID-FILE                         VALUE 36.
        88  AS-INVALID-BACKWARD                     VALUE 38.
        88  AS-INVALID-INPUT-LENGTH                 VALUE 40.
        88  AS-CORRESPONDENT-REJECT                 VALUE 42.
        88  AS-NAD-REJECT                           VALUE 43.
        88  AS-INVALID-TIMOUT                       VALUE 44.
        88  AS-OPEN-OVERFLOW                        VALUE 45.
        88  AS-NO-TELECOMMUNICATION                 VALUE 46.
        88  AS-INVALID-MODE                         VALUE 47.
        88  AS-SATURATED-CORRESPONDENT              VALUE 50.
        88  AS-SECURITY-CHECK-FAILED                VALUE 51.
        88  AS-SYSTEM-CONDITION                     VALUE 52.
        88  AS-DUPNAME                              VALUE 53.
        88  AS-SESSION-SHUTDOWN                     VALUE 54.
        88  AS-DUPLICATE-FILTER                     VALUE 61.
      03    AS-MSG-STATUS              COMP-1       VALUE 0.
        88  AS-END-OF-REGION                        VALUE 1.
      03    AS-SYSTEM-STATUS           COMP-2       VALUE 0.
      03    AS-AUX-STATUS              COMP-1       VALUE 0.
      03    AS-ERROR-ARG-INDEX         COMP-1       VALUE 0.
    02    AS-TIMOUT                    COMP-1       VALUE 0.
    02    AS-CNX-TYPE                  PIC X(2)     VALUE " ".
        88  AS-INITIATE                             VALUE "IN".
        88  AS-ACCEPT                               VALUE "AC".
        88  AS-INPUT-FILE                           VALUE "IF".
        88  AS-OUTPUT-FILE                          VALUE "OF".
    02    AS-ACID                      COMP-2       VALUE 0.
    02    AS-BACKWARD                  PIC X        VALUE "F".
        88  AS-FORWARD-LOGIC                        VALUE "F".
        88  AS-BACKWARD-LOGIC                       VALUE "B".
        88  AS-ABSOLUTE-LOGIC                       VALUE "A".
        88  AS-RELATIVE-LOGIC                       VALUE "R".
        88  AS-END-LOGIC                                VALUE "E".
        88  AS-CONTINUATION-LOGIC                       VALUE " ".
    02    AS-FILLER-0                     PIC X         VALUE " ".
    02    AS-OPEN-INFO.
      03    AS-NSTD                       PIC X(2)      VALUE " ".
        88  AS-FILE-LOG                                 VALUE "FL".
        88  AS-DSAC-LOG                                 VALUE "LG".
        88  AS-SESSION                                  VALUE "SS".
        88  AS-LOOPBACK                                 VALUE "LP".
      03    AS-LOCAL-MAILBOX.
        04    AS-L-SCID                   PIC X(4)      VALUE " ".
        04    AS-L-MBX                    PIC X(8)      VALUE " ".
        04    AS-L-EXT                    PIC X(4)      VALUE " ".
      03    AS-DEST-MAILBOX.
        04    AS-D-SCID                   PIC X(4)      VALUE "ANY ".
        04    AS-D-MBX                    PIC X(8)      VALUE "$NAD".
        04    AS-D-EXT                    PIC X(4)      VALUE " ".
      03    AS-SUBMITTER-ID.
        04    AS-BILLING                  PIC X(12)     VALUE " ".
        04    AS-PERSON                   PIC X(12)     VALUE " ".
        04    AS-PROJECT                  PIC X(12)     VALUE " ".
        04    AS-PASSWORD                 PIC X(12)     VALUE " ".
      03    AS-WINDOW-SIZE                COMP-1        VALUE 32767.
      03    AS-PATHNAME                   PIC X(200)    VALUE " ".
    02    AS-RECORD-NUMBER                COMP-2        VALUE 1.
```

7.1.6 AS-AUPI-SYSTEM

Function

To declare the AUPI system information data structure describing a given system. This structure is used by H_NA_IGETINF.

Format

```
COPY AS-AUPI-SYSTEM.
```

Expansion of the Primitive: COPY AS-AUPI-SYSTEM

```
01    AS-AUPI-SYSTEM .
  02    AS-REVISION          COMP-1            VALUE 0.
  02    AS-SYST-NAME         PIC X(4)          VALUE " ".
  02    AS-MACH-NAME         PIC X(8)          VALUE " ".
  02    AS-OPER-SYST         PIC X(8)          VALUE " ".
  02    AS-RELEASE           PIC X(8)          VALUE " ".
  02    AS-SYST-TYPE         PIC X             VALUE " ".
  02    AS-FILLER-0          PIC X             VALUE " ".
```

7.2 Procedure COBOL Calls

7.2.1 H_NA_ICLCR

Close-Correspondent

Function

To close the relationship existing between the AUT and the Administrative Correspondent.

Format

```
CALL "H_NA_ICLCR" USING AS-AUPI-STATUS.
```

7.2.2 H_NA_ICREFL

Create-Filter

Function

To associate a filter with an Administrative Correspondent.

Format

```
CALL "H_NA_ICREFL" USING AS-AUPI-STATUS, AF-AUPI-FILTER.
```

7.2.3 H_NA_IDELFL

Delete-Filter

Function

To delete the filter associated with an Administrative Correspondent.

Format

```
CALL "H_NA_IDELFL" USING AS-AUPI-STATUS, AF-AUPI-FILTER.
```

7.2.4 H_NA_IGETFD

Receive-Field

Function

To receive a specific field of an administrative message according to the region and the access mode.

Format

```
CALL "H_NA_IGETFD" USING AS-AUPI-STATUS, AF-AUPI-FIELD.
```

7.2.5 H_NA_IGETHD

Receive-Header

Function

To get the header of an administrative message.

Format

```
CALL "H_NA_IGETHD" USING AS-AUPI-STATUS, AH-AUPI-HEADER [,TIME].
```

Description of Parameters

TIME:	receives the time in the format: 'YYMMDDHHMNSSCC'

7.2.6 H_NA_IGETINF

Get-System-Information

Function

To get information describing a system.

Format

```
CALL "H_NA_IGETINF" USING AS-AUPI-STATUS, AS-AUPI-SYSTEM.
```

7.2.7 H_NA_IGETPOS

Set-Position

Function

To position the current pointer of the AUT on the current pointer of the ASF log function.

Format

```
CALL "H_NA_IGETPOS" USING AS-AUPI-STATUS.
```

7.2.8 H_NA_IGETREC

Receive-Record

Function

To receive a specific AEP record according to the filters.

Format

```
CALL "H_NA_IGETREC" USING AS-AUPI-STATUS, AR-AUPI-RECORD.
```

7.2.9 H_NA_IGETVB

Receive-Verbatim

Function

To get a verbatim AEP record.

Format

```
CALL "H_NA_IGETVB" USING AS-AUPI-STATUS, AF-AUPI-FIELD.
```

7.2.10 H_NA_IOPCR

Open-Correspondent

Function

To open a relationship between the AUT and the Administrative Correspondent.

Format

```
CALL "H_NA_IOPCR" USING AS-AUPI-STATUS [AS-MODE [AS-SESSION]].
```

AS-MODE: optional input parameter specifying the mode, see Appendix B in the *AUPI User's Guide*. The default is AUPI.

```
77 AS-MODE PIC X(4).
88 AEP-MODE  VALUE "AEP".
88 AUPI-MODE VALUE "AUPI".
```

AS-SESSION: optional input parameter specifying the maximum number of sessions that can be simultaneously opened. The default is 50.

```
77 AS-SESSION USAGE IS COMP-1.
```

7.2.11 H_NA_ISENDFD

Send-Field

Function

To send a specific field of the administrative message according to the region and the access mode.

Format

```
CALL "H_NA_ISENDFD" USING AS-AUPI-STATUS, AF-AUPI-FIELD.
```

7.2.12 H_NA_ISENHD

Send-Header

Function

To build and, ultimately, to send an AEP command header according to the input parameters supplied.

Format

```
CALL "H_NA_ISENDHD" USING AS-AUPI-STATUS, AH-AUPI-HEADER.
```

7.2.13 H_NA_ISENDVB

Send-Verbatim

Function

To send a verbatim AEP record to the Loopback Correspondent.

Format

```
CALL "H_NA_ISENDVB" USING AS-AUPI-STATUS, AF-AUPI-FIELD.
```

❑

8. Generalized Terminal Writer (GTWriter)

For a detailed description of how to use GTWriter procedures, refer to the *GTWriter User's Guide*.

This chapter deals with the following procedures COBOL calls:

H_TW_UCOMM	Sends a GTWriter command to the Command Handler.
H_TW_UDRE	Returns the status of a specified driver.
H_TW_UFORM	Returns the structure describing a form.
H_TW_UGETR	Returns the allocated report number.
H_TW_UMAINE	Reads fields in the main GTWriter table.
H_TW_UPOOL	Returns the structure describing a pool.
H_TW_UQNE	Reads the next report in the GTWriter queue.
H_TW_UQRE	Reads a report description from the GTWriter queue.
H_TW_USAVE	Saves a report member in the SITEOUT library.
H_TW_USTARTE	Opens a report.
H_TW_UTRE	Returns a description of a terminal and its state.
H_TW_UUSER	Returns the structure describing a user.

8.1 H_TW_UCOMM

Syntax

```
CALL "H_TW_UCOMM" USING TW-COMMAND TW-TEXT TW-LENGTH TW-RESULT.
```

Description

Sends a GTWriter command to the Command Handler.

The COBOL statement `COPY TW-COMM-AREA` inserts the following structure in the program at compilation time.

```
01  TW-COMMAND     PIC X 4).
01  TW-TEXT        PIC X(179).
01  TW-LENGTH      COMP-1.
```

For reasons of compatibility, you can continue to specify a size of 200 characters for the TW-TEXT (no need to recompile), but only the first 179 characters are taken into account.

TW-COMMAND and TW-TEXT are the name and parameters, respectively, of a OCL command.

TW-LENGTH indicates the length of this text.

TW-RESULT is COMP-2 and is set to the following:

0	= done
1	= H_TWCOMM not running
3	= CDUNKN: command unknown
4	= invalid length
5	= erroneous parameter(s)
-1	= unable to send command to command handler

NOTE:
Only the OCL commands are authorized. See Appendix B in the *GTWriter User's Guide*.

8.2 H_TW_UDRE

Syntax

```
CALL "H_TW_UDRE" USING TW-DRIV-DESC TW-RESULT.
```

Description

Returns the status of a specified Driver.

The COBOL statement COPY TW-DRIVER-AREA inserts the following structure in the program at compilation time.

```
  01  TW-DRIV-DESC.
      03  TW-DRIV-NUMB         COMP-1.
      03  TW-DRIV-NAME         PIC X(4).
      03  TW-DRIV-RON          PIC X(5).
      03  TW-DRIV-STAT         COMP-1.
          88 STOPPED           VALUE 1.
          88 KNOWN             VALUE 2.
          88 STARTED           VALUE 3.
          88 ACTIVE            VALUE 4.
          88 IN-ABORT          VALUE 5.
          88 ABORTED           VALUE 6.
      03  TW-DRIV-TERMS        COMP-1.
      03  TW-DRIV-CONN         COMP-1.
      03  TW-DRIV-PRINT        COMP-1.
      03  TW-DRIV-MAX          COMP-1.
      03  TW-DRIV-MOUNTS       COMP-1.
      03  TW-DRIV-CONNECTING   COMP-1.
      03  TW-DRIV-RECOVERS     COMP-1.
      03  TW_DRIV-TYPE         COMP-1.
          88 DRIV-IS-NORMAL    VALUE 1.
          88 DRIV-IS-MATHILDE  VALUE 2.
      03  FILLER               PIC X(18).
```

TW-RESULT is COMP-2 and is set to the following:

0	= done
1	= INDOUT: wrong Driver number
2	= GTWriter not generated
3	= WRGDRIV: wrong Driver name
5	= erroneous parameter(s)

TW-DRIV-STAT: driver’s current status:

1	= stopped
2	= known
3	= started
4	= active
5	= aborting
6	= aborted

8.3 H_TW_UFORM

Syntax

```
CALL "H_TW_UFORM" USING TW-FORM-DESC TW-RESULT.
```

Description

Returns the structure describing a form.

The COBOL statement COPY TW-FORM-AREA inserts the following structure in the program at compilation time:

```
************************************************
*            TWRITER FORM DESCRIPTION          *
************************************************
 01  TW-FORM_DESC.
     03   TW-FORM-NAME         PIC X(6)
     03   TW-FORM-HT           COMP-1.
     03   TW-FORM-TOP          COMP-1.
     03   TW-FORM-BOTTOM       COMP-1.
     03   TW-FORM-PTLINES      COMP-1.
     03   TW-FORM-PTEST        OCCURS 4.
          04  TW-FORM-LINENO       COMP-1.
          04  TW-FORM-COL          COMP-1.
          04  TW-FORM-TEXT         PIC X(40).
     03   TW-FORM-PAPER        PIC X(6).
     03   TW-FORM-ENV          PIC X(6).
     03   TW-FORM-TYPE         COMP-1.
          88  FORM-IS-NORMAL   VALUE 1.
          88  FORM-IS-STREAM   VALUE 2.
     03   TW-FORM-WIDTH        COMP-1.
     03   TW-FORM-EJECT        COMP-1.
          88  NO-EJECT         VALUE 0.
          88  EJECT-IS-PAGE    VALUE 1.
          88  EJECT-IS-END     VALUE 2.
     03   FILLER               PIC X(2).
```

TW-RESULT is COMP-2 and is set to the following:

0	= done
2	= GTWriter is not generated
3	= unknown form name
5	= erroneous parameter (s)

8.4 H_TW_UGETR

Syntax

```
CALL "H_TW_UGETR" USING TW-REPORTNB.
```

Description

Returns the report number allocated by GTWriter when the CALL "H_TW_USTARTE" procedure is called.

TW-REPORTNB is COMP-1 and will contain the last allocated report number for this user; if H_TW_USTART was not called, then 0 is returned.

8.5 H_TW_UMAINE

Syntax

```
CALL "H_TW_UMAINE" USING TW-MAIN-DESC TW-RESULT.
```

Description

Reads fields declared at GTWriter generation in the main table of GTWriter.

The COBOL statement COPY TW-MAIN-AREA inserts the following structure in the program at compilation time.

```
01 TW-MAIN-DESC.
  03 TW-MAIN-SITE        PIC X(20).   "site-description" of GEN
                                      statement
  03 TW-MAIN-DEFTERMS    COMP-1.      number of TERM statements
  03 TW-MAIN-DEFUSERS    COMP-1.      number of USER statements
  03 TW-MAIN-DEFPOOLS    COMP-1.      number of POOL statements
  03 TW-MAIN-DEFFORMS    COMP-1.      number of FORM statements
  03 TW-MAIN-CLASS       PIC X.       CLASS parameter of GEN
  03 TW-MAIN-REVISION    PIC X(5).    current Version of GTWriter
  03 TW-MAIN-DEFDRIVERS  COMP-1.      total number of drivers as
                                      referenced in the DRIVER
                                      parameters of all
                                      TERM statements
  03 TW-MAIN-QUEUE       COMP-1.      total number of reports
                                      known to the system
  03 FILLER              PIC X(20).
```

TW-RESULT is COMP-2 and is set to the following:

0	= done
1	= GTWriter not generated
5	= erroneous parameter(s)

8.6 H_TW_UPOOL

Syntax:

```
CALL "H_TW_UPOOL" USING TW-POOL-DESC TW-RESULT.
```

Description

Returns the structure describing a pool.

The COBOL statement COPY TW-POOL-AREA inserts the following structure in the program at compilation time:

```
*************************************
*     TWRITER POOL DESCRIPTION      *
*************************************
 01  TW-POOL-DESC.
     03  TW-POOL-NAME           PIC X(12)
     03  TW-POOL-DRIVER         COMP-1
     03  TW-POOL-OWNER          PIC X(12)
     03  TW-POOL-AUTO           PIC X(8)
     03  TW-POOL-QUEUE          COMP-1
     03  TW-POOL-SIZE           COMP-1
     03  TW-POOL-LIST           OCCURS 16
         04 TW-POOL-TERM-NAME   PIC X(12)
         04 TW-POOL-TERM-NUMB   COMP-1
     03  FILLER                 PIC X(128)
```

TW-RESULT is COMP-2 and is set to the following:

0	= done
1	= GTWriter is not generated
2	= ISTERM: the given name is the term
3	= WRGPOOL: wrong pool name
4	= erroneous parameter (s)
-1	= error while accessing terminal list
-2	= error while accessing pool list

8.7 H_TW_UQNE

Syntax

```
CALL "H_TW_UQNE" USING TW-REPORT-DESC TW-RESULT.
```

Description

Reads the next report of the GTWriter queue. It is called after H_TW_UQRE.

The COBOL statement COPY TW-REPORT-AREA inserts the same structure as for the CALL "H_TW_UQRE" procedure in the program at compilation time.

TW-RESULT is COMP-2 and is set to the following:

0	= done
1	= GTWriter not generated
2	= INDOUT: wrong report number (>4999 or <1)
3	= WRGREP: wrong report number (= -1)
4	= DATALIM: no next report in the queue
5	= erroneous parameter(s)
-1	= wrong return code from H_SYSPUT
-2	= wrong return code from H_SYSGET
-3	= error while accessing the report queue
-4	= error while accessing pool list
-5	= error while accessing terminal list

8.8 H_TW_UQRE

Syntax

```
CALL "H_TW_UQRE" USING TW-REPORT-DESC TW-RESULT.
```

Description

Reads a report description from the GTWriter report queue.

The report is stored in the TW-REPORT-DESC structure. The COBOL statement COPY TW-REPORT-AREA inserts the structure (given on the next page) in the program at compilation time.

TW-RESULT is COMP-2 and is set to the following:

0	=	done
1	=	GTWriter is not generated
2	=	INDOUT: wrong report number (>5000 or <1)
3	=	WRGREP: unknown report
5	=	erroneous parameter(s)
-1	=	error while accessing the report queue
-2	=	error while accessing the report queue
-3	=	error while accessing the report queue
-4	=	error while accessing pool list
-5	=	error while accessing terminal list

```
******************************************************
*             TWRITER REPORT DESCRIPTION AREA        *
******************************************************
 01  TW-REPORT-DESC.
     03  TW-RD-NUMBER                COMP-1
     03  TW-RD-CLASS                 PIC X.
     03  TW-RD-PRTY                  COMP-1.
     03  TW-RD-TERM                  PIC X(12).
     03  TW-RD-USER                  PIC X(12).
     03  TW-RD-STATE                 COMP-1.
         88  REPORT-IS-FROZEN        VALUE 1.
         88  REPORT-IS-HELD          VALUE 2.
         88  REPORT-IS-WAITING       VALUE 3.
         88  REPORT-IS-PRINTING      VALUE 4.
         88  REPORT-IS-CONNECTING    VALUE 5.
         88  REPORT-IS-MOUNTING      VALUE 6.
     03  TW-RD-FILEDEF               COMP-1.
         88  REPORT-IN-SYSOUT        VALUE 1.
         88  REPORT-IN-SITEOUT       VALUE 2.
         88  REPORT-IN-FILE          VALUE 3.
         88  REPORT-IN-MEMBER        VALUE 4.
     03  TW-RD-FSTAT                 COMP-1.
         88  FSTAT-CAT               VALUE 1.
         88  FSTAT-RESIDENT          VALUE 2.
         88  FSTAT-NOT-GIVEN         VALUE 3.
     03  TW-RD-EFN                   PIC X(44).
     03  TW-RD-MD                    PIC X(6).
     03  TW-RD-DVC                   PIC X(20).
     03  TW-RD-FSN                   COMP-1.
     03  TW-RD-MB                    PIC X(31).
     03  TW-RD-PROJECT               PIC X(12).
     03  TW-RD-RON                   PIC X(5).
     03  TW-RD-SUBMITTER             PIC X(12).
     03  TW-RD-FORM                  PIC X(6).
     03  TW-RD-BANNER                COMP-1.
         88  BANNER-NOT-SPECIFIED    VALUE 1.
         88  BANNER-REQUESTED        VALUE 2.
         88  NO-BANNER-REQUESTED     VALUE 3.
     03  TW-RD-DELETE                COMP-1.
         88  DELETE-REPORT           VALUE 1.
     03  TW-RD-DATAFORM              COMP-1.
         88  DATAFORM-UNKNOWN        VALUE 1.
         88  DATAFORM-SARF           VALUE 2.
         88  DATAFORM-SSF            VALUE 3.
         88  DATAFORM-ASA            VALUE 4.
         88  DATAFORM-VPF            VALUE 5.
     03  TW-RD-RESTART               COMP-1.
         88  NO-RESTART              VALUE 1.
         88  RESTART-FORWARD         VALUE 2.
```

```
        88  RESTART-BACK            VALUE 3.
        88  RESTART-PAGE            VALUE 4.
    03  TW-RD-RESTPAGES             COMP-1.
    03  TW-RD-COPIES-ASKED          COMP-1.
    03  TW-RD-COPIES-DONE           COMP-1.
    03  TW-RD-LINES                 COMP-2.
    03  TW-RD-CURPAGE               COMP-1.
    03  TW-RD-CREATION-DATE.
        04  TW-RD-CR-YY             COMP-1.
        04  TW-RD-CR-MT             COMP-1.
        04  TW-RD-CR-DD             COMP-1.
    03  TW-RD-CREATION-TIME.
        04  TW-RD-CR-HH             COMP-1.
        04  TW-RD-CR-MN             COMP-1.
        04  TW-RD-CR-SC             COMP-1.
    03  TW-RD-NAME                  PIC X(06).
```

8.9 H_TW_USAVE

Syntax

```
CALL "H_TW_USAVE" USING GTWRITER-FILE TW-NUMBER TW-USER
                        TW-REPORT-NAME TW-RESULT.
```

Description

This procedure is used by a TPR to save a report member in the library SITEOUT. It must be called after the commitment point is taken.

The report member must have been created by H_TW_USTART in the HOLD state.

TW-NUMBER	is a COMP-1 field that contains the report number allocated to the report.
TW-USER	is a PIC(12) field that contains an identifier. It is used to create the name of the member in the SITEOUT file.
TW-REPORT-NAME	is a PIC(18) field that returns the name of the member in SITEOUT. This name is obtained by concatenating the names of the user and the report number (for example, USERA_R18).
TW-RESULT	is a COMP-2 field and has the following possible values:

- 0 = done
- 1 = GTWriter is not generated
- 2 = report number is unknown in SYS.TW.OUT
- 3 = report unknown
- 4 = file SITEOUT does not exist
- 5 = SITEOUT library is saturated
- 6 = file is not assigned to SYS.TW.OUT
- 7 = report is not in the HOLD state
- -1 = error occurred while processing subfile.

8.10 H_TW_USTARTE

Syntax:

```
CALL "H_TW_USTARTE" USING TWRITER-FILE TW-REPORT-NAME TW-RESULT.
                          [TW-REPORTNB]
```

Description:

H_TW_USTARTE procedure is called in a TPR to open report.

The COBOL statement COPY TW-INTERFACE inserts the following input structure in the program at compilation time.

```
  01  TW-RESULT        COMP-2.
  01  TW-REPORT
      03  TW-DEST      PIC X(12).
      03  TW-FORM      PIC X(6).
      03  TW-NAME      PIC X(6).
      03  TW-CLASS     PIC X.
      03  TW-BANNER    PIC X.
      03  TW-SCHED     PIC X.
      03  TW-SILENT    PIC X.
      03  TW-COPIES    PIC 99.
      03  TW-PRTY      PIC 9.
      03  TW-USER      PIC X(12).
      03  TW-FILLER    PIC X(25).
```

Description of Fields:

TW-DEST	destination terminal or pool; if blank then the default is used if possible
TW-FORM	name of the paper to be used; if blank then the default is used
TW-NAME	name of the report; if blank then the report has no name
TW-CLASS	class of the report; if blank then the default is used
TW-BANNER	Y means produce banners, N means no banners; if blank, the terminal default is used.

TW-SCHED	N means hold, Y means do not hold.
TW-SILENT	N means no confirmation is required of outputs being started or completed, Y means confirmation is required.
TW-COPIES	number of copies required; if blank, 1 is assumed.
TW-PRTY	priority from 0-7 must be given.
TW-USER	name of the USER.

TW-RESULT contains the result of the call; the values are as follows:

0	= done
1	= file is not assigned to SYS.TW.OUT
2	= destination is not known
3	= form is not known
4	= copies are not numeric or out of range
5	= class is not A-Z or space
6	= destination is blank but the user has not default terminal
7	= banner not equal Y, N or space
8	= non-concurrent Note that this value tests some class of non-concurrency, but not all. Therefore, you should nor use it as an indicator
9	= invalid priority
10	= GTWriter generation not performed
11	= 5000 reports already exist in the queue
12	= USTART already called in the commitment unit
13	= error in parameter
-2	= wrong return code from H_OPENS Note that this return code is sent if the maximum number of subfiles (reports) simultaneously open is reached. You can modify this number at TDS generation using the "USE TWRITER-NUMDF-xxx" clause (where xxx has a value ranging from 001 to 200).
-3	= wrong return code from H_PUT (control record 101)
-4	= error while accessing pool list
-5	= error while accessing terminal list

TW-REPORTNB contains the report number and is a COMP-1 field.

H-TW-USTARTE creates a control record. The report will contain everything written to the file from now to the commitment or to the end of the transaction. If you wish to create a further report, you must include CALL "H_TW_USTARTE" in another commitment unit.

NOTE:

"TWRITER-FILE" in the calling sequence means the COBOL file-name of the TDS non-controlled file assigned to SYS_TW_OUT.

If a abort occurs during a commitment unit, the report is not enqueued and the commitment unit is restarted. The CALL "H_TW_USTARTE" statement can be called again because the subfile is opened in output mode.

8.11 H_TW_UTRE

Syntax

```
CALL "H_TW_UTRE" USING TW-TERM-DESC TW-RESULT.
```

Description

Returns the description of a terminal and its state. The terminal is identified by its logical name contained in the field TW-TERM-NAME.

If this field is equal to "space", the terminal is identified by its physical name that is by TW-TERM-NODE and TW-TERM-ID fields.
If these latter fields are equal to "space", then the terminal is identified by its number contained in the field TW-TERM-NUMBER.

The COBOL statement `COPY TW-TERM-AREA` inserts the following data structure in the TPR at compilation time.

```
    01  TW-TERM-DESC.
        03  TW-TERM-NAME                       PIC X(12).
        03  TW-TERM-NODE                       PIC X(8).
        03  TW-TERM-ID                         PIC X(8).
        03  TW-TERM-NUMB                       COMP-1.
        03  TW-TERM-DRIVER                     COMP-1.
        03  TW-TERM-FORM                       PIC X(6)
        03  TW-TERM-FORM-MOUNT                 PIC X(6)
        03  TW-TERM-BLOCKING                   COMP-1.
        03  TW-TERM-REPORT                     COMP-1.
        03  TW-TERM-ASG                        PIC X(12).
        03  TW-TERM-PADDING                    COMP-1.
        03  TW-TERM-RETRYCT                    COMP-1.
        03  TW-TERM-CLASSES                    PIC X  OCCURS 26.
        03  TW-TERM-OWNER                      PIC X(12).
        03  TW-TERM-INVCHAR                    PIC X.
        03  TW-TERM-MANUAL                     COMP-1.
    88            TERM-IS-AUTO                 VALUE 1.
    88            TERM-IS-MANUAL               VALUE 2.
        03  TW-TERM-KEEP                       COMP-1.
    88            TERM-IS-KEEP                 VALUE 1.
    88            TERM-IS-NOT-KEEP             VALUE 2.
        03  TW-TERM-MOUNT                      COMP-1.
    88            ASK-FOR-MOUNTS               VALUE 1.
    88            WAIT-FOR-MOUNTS              VALUE 2.
        03  TW-TERM-STATUS                    COMP-1.
    88            TERM-IS-IDLE                VALUE 1.
    88            TERM-IS-CONNECTING          VALUE 2.
    88            TERM-IS-MOUNTING            VALUE 3.
    88            TERM-IS-PRINTING            VALUE 4.
    88            TERM-IS-RECOVERING          VALUE 5.
    88            TERM-IS-CLOSED              VALUE 6.
    88            TERM-NOT-CONNECTED          VALUE 7.
        03  TW-TERM-QUEUE                     COMP-1.
        03  TW-TERM-MODEL                     COMP-1.
        03  TW-TERM-REJECT-CODE               COMP-1.
        03  TW-TERM-MAXCONCT                  COMP-1.
        03  TW-TERM-CONNECTCT                 COMP-1.
        03  TW-TERM-TYPE                      COMP-1.
    88            TERM-IS-NIP3                VALUE 1.
    88            TERM-IS-NORMAL              VALUE 2.
    88            TERM-IS-STREAM              VALUE 3.
    88            TERM-IS-MATHILDE            VALUE 4.
    88            TERM-IS-REMOTE              VALUE 5.
        03  TW-TERM-REPEATMSG                 COMP 1.
    88            TERM-IS-REPEATMSG           VALUE 1.
    88            TERM-IS-NOT-REPEATMSG       VALUE 2.
        03  TW-TERM-NOCONSKIP                 COMP 1.
    88            TERM-IS-NOCONSKIP           VALUE 1.
    88            TERM-IS-NOT-NOCONSKIP       VALUE 2.
        03  TW-TERM-REALSKIP                  COMP 1.
    88            TERM-IS-REALSKIP            VALUE 1.
    88            TERM-IS-NOT-REALSKIP        VALUE 2.
        03  TW-TERM-CSET                      COMP-1
        03  TW-TERM-FORMLOCK                  COMP-1.
    88            FORMLOCK-AND-NOT-ACCOUNT    VALUE 1.
    88            NOT-FORMLOCK-AND-NOT-ACCOUNT VALUE 2.
    88            FORMLOCK-AND-ACCOUNT        VALUE 3.
    88            NOT-FORMLOCK-AND-ACCOUNT    VALUE 4.
```

TW-RESULT is COMP-2 and is set to the following:

0	=	done
1	=	GTWriter is not generated
2	=	ISPOOL: the given name is a pool name
3	=	WRGTERM: wrong terminal name
4	=	INDOUT: wrong terminal number
5	=	erroneous parameter(s)
-1	=	error while accessing terminal list
-2	=	error while accessing pool list

8.12 H_TW_UUSER

Syntax

```
CALL "H_TW_UUSER" USING TW-USER-DESC TW-RESULT.
```

Description

Returns the structure describing a user.

The COBOL statement COPY TW-USER-AREA inserts the following structure in the program at compilation time.

```
*******************************************
*       TWRITER USER DESCRIPTION          *
*******************************************
 01   TW-USER-DESC.
      03 TW-USER-NAME           PIC X(12)
      03 TW-USER-TERM           PIC X(12)
      03 TW-USER-CLASS          PIC X(1)
      03 TW-USER-PRIORITY       COMP-1.
      03 TW-USER-PRILIM         COMP-1.
      03 TW-USER-MASTER         COMP-1.
      03 TW-USER-CONTROL        COMP-1.
      03 FILLER                 PIC X(20)
```

TW-RESULT is COMP-2 and is set to the following:

0	= done
2	= GTWriter is not generated
3	= unknown user name
5	= erroneous parameter(s)
-1	= error while accessing pool list

9. Unified File Transfer (UFT)

For a detailed description of how to use UFT procedures, refer to the *UFT User’s Guide*.

This chapter deals with the following COBOL procedure calls:

H_NP_UCANFT	Cancel a file transfer request.
H_NP_UHLDFT	Hold a file transfer request.
H_NP_URELFT	Release a file transfer request.
H_NP_USUBFT	File transfer request.
H_NP_UTESTFT	Status of a file transfer request.

9.1 Cancel/Hold/Release Interface (H_NP_UCANFT, H_NP_UHLDFT, H_NP_URELFT)

Programmatic Format

```
CALL "H_NP_UCANFT or H_NP_UHLDFT or H_NP_URELFT"
      USING SUBFT-EXECUTION-RESULT.
```

Structure of SUBFT-EXECUTION-RESULT

The following COBOL structure will receive values giving the Results of the Execution.

```
   01    SUBFT-EXECUTION-RESULT.
     02    SUBFT-RESULT          COMP-2.
     02    REQUEST-INDEX         COMP-2.
```

SUBFT-RESULT

output parameter: a number to identify the result of the action (cancel, hold or release) requested in the programmatic interface.

0 = the request has been found and the action performed.

1 = the request was not found which could imply that the given request index was wrong.

2 = the request exists but the given index does not correspond to a File Transfer request.

3 = several requests have the same number.

This is normally not possible since it is the request index which is given in input and not a request name.

4 = the request does not belong to the submitter of the CALL.

This is an access right violation.

5 =	the request has already been held if HOLD was issued.
6 =	the request has already be released if RELEASE was issued.

NOTE:
The G4 general register status register is set as follows to:

- DONE if SUBFT_RESULT is 0
- OPTERR if SUBFT_RESULT is other than 0.

REQUEST-INDEX	*input parameter:* the file transfer internal index allocated by the DJP/UFT queue manager, on which the action was performed.

9.2 File Transfer Request Interface (H_NP_USUBFT)

Programmatic Format

```
CALL "H_NP_USUBFT" USING COBOL-SUBFT-PARAMETERS,
     <infile_literal>,
     <outfile_literal>,
     SUBFT-EXECUTION-RESULT.
     [, TDS-COMMITMENT-IDENTIFICATION]
     [, INDEF-PARAMETERS]
     [, OUTDEF-PARAMETERS]
     [, OUTALC-PARAMETERS]
```

Structure of COBOL-SUBFT-PARAMETERS

An initialized COBOL structure describes the file transfer request parameters.

```
   01    COBOL-SUBFT-PARAMETERS.
     02    SUBFT-LENGTH                     COMP-1.
     02    QUEUING-PARAMETERS.
       03    REQUEST-NAME                   PIC X(8).
       03    REQUEST-PRIORITY               PIC X(1).
       03    REQUEST-HOLD                   PIC X(1).
       03    REQUEST-CLASS                  PIC X(1).
       03    WAIT-DATE-TIME.
         04    WHEN-PARAMETER.
           05    WHEN-LENGTH                COMP-1.
           05    WHEN-VALUE                 PIC X(32).
         04    REPEAT-PARAMETER.
           05    REPEAT-LENGTH              COMP-1.
           05    REPEAT-VALUE               PIC X(7).
         04    EVERY-PARAMETER.
           05    EVERY-LENGTH               COMP-1.
           05    EVERY-VALUE                PIC X(12)
         04    RST-PARAMETER.
           05    RST-LENGTH                 COMP-1.
           05    RST-VALUE                  PIC X(32).
      03    WAIT-FILE-TRANSFERS.
         04    WAIT-FT-NUMBER               COMP-1.
         04    WAIT-FT-REQ-LIST.
           05    WAIT-FT-REQID   OCCURS(8)  PIC X(8).
     02    FILE-TRANSFER-OPTIONS.
       03    OPT-BINARY                      PIC X(1)
       03    NCOMPACT                        PIC X(1).
       03    APPEND                          PIC X(1).
       03    RESTART                         PIC X(1)
       03    PASSWORD                        PIC X(1).
       03    BRIEF                           PIC X(1).
       03    TRACE                           PIC X(1).
       03    DELETE                          PIC X(1).
     02    RFU-FIELDS                        PIC X(22).
```

SUBFT-LENGTH contains the length of the COBOL-SUBFT-PARAMETERS structure and identifies its version. Length is 200 characters, RST is introduced as a new parameter, PASSWORD, BRIEF, TRACE introduced as new file transfer options and RFU-FIELDS added.

QUEUING-PARAMETERS
a set of parameters for the UFT queue manager.

REQUEST-NAME:
identifies an external request name; must be set to space if request name is not specified.

REQUEST-PRIORITY:
represents the priority of the request for selection purposes from 0 through 7, 0 being the highest, 7 being the lowest. The default request priority is 3.

REQUEST-HOLD:
0=execute the file transfer request.
1=hold the request in UFT queue. It will be released for execution:
either by RUR (Release User Request)
or by a call to procedure H_NP_URELFT.

REQUEST-CLASS:
represents the class of the request from A through Z, used by the selection mechanism. In V2, UFT requests are selected by priority. The class criteria is not taken into account by the queue manager. If this field is blank, the default request class C is assumed. Note that if request_class is specified and request_priority is not specified, the affected priority is deduced from the request_class:
A-->1, B-->2, C-->3, D-->4, E-->5, F-->6, G-Z-->7.

WAIT-DATE-TIME:
it is possible to submit a file transfer request and to launch its execution at a given date-time, or after a given time, and to ask for repetitive periodic executions of the request.

Four parameters may be specified: the when-parameter, repeat-parameter, every-parameter and rst-parameter.

The actual lengths of these parameters must be set in the corresponding when-length, repeat-length, every-length and rst-length fields.

The length field must be equal to 0 when the corresponding parameter is not specified. The value fields when-value, repeat-value, every-value and rst-value are alphanumeric and must follow the syntax:

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

This parameter specifies the date_time for request execution.

IMMED means immediate execution (default value).

Absolute date_time:

```
mm = month, dd = day, yy = year
hh = hour, mm = minutes
```

When only hh.mm is specified, date of submission day is assumed.

Relative time:

```
+ddddd = 1 to 99999

W = week, D = day, H = hour, M = minutes.

REPEAT = { NO | FOREVER | ddddd }
```

This parameter specifies the number of repetitions of request execution.

```
NO      = no repetition (default value)

FOREVER = infinite repetition:
            default when EVERY is specified

ddddd   = number of repetitions
```

```
        { {W|WEEK} | {D|DAY} | {H|HOUR} }
EVERY = {                                }
        { ddddd { W | D | H | M }        }
```

This parameter specifies the period between two executions of the request. It is mandatory when REPEAT is specified. In the case of minutes (dddddM) this period must be greater than 15.

```
         { IMMED       }
RST =    { CANCEL | NO }
         {<delay> [E]  }
```

This parameter specifies how the file transfer request should be handled in case of system failure or shutdown, or if the system was not running at normal execution time.

IMMED (default): the request is enqueued

NO or CANCEL: the request is canceled

<delay>: expressed as <decimal>{M|H|D|W} is the delay added to the normal queuing time where: M=month, H=hour, D=day, W=week.

E: option to <delay> is the elapsed delay before which, on system restart, the request is enqueued and executed.

WAIT-FILE-TRANSFERS:
allows executing the submitted request on completion of up to 8 other file transfer requests.

WAIT-FT-NUMBER:
must be set to 0, if no request is to wait.

WAIT-FT-REQ-LIST:
identifies the corresponding file transfer requests. The identifier in WAIT-FT-REQID is either a FON or the NAME of EFTR command or the REQUEST-NAME in the COBOL-SUBFT-PARAMETERS structure.

FILE-TRANSFER-OPTIONS
defines the conditions of the file transfer.

OPT-BINARY:
0=character files.
1=binary files (records are transferred without any modification). In SSF files, record headers and control records are transferred if OPT-BINARY=1 or removed before transfer if OPT-BINARY=0.

NCOMPACT:
0=data compaction is performed before data transfer.
1=no data compaction is performed before data transfer.

APPEND:
0=output file is opened in OUTPUT mode.
1=output file is opened in APPEND mode.

RESTART:
0=the input file is totally transferred.
1=specifies that the request was interrupted during its execution.

PASSWORD:
0=request submitter's password is not given.
1=request submitter's password is given.

BRIEF:
0=normal set of messages are delivered from the beginning and up to the end of the transfer.
1=a very brief set of messages is given (this parameter has the same purpose as the BRIEF parameter of the EFTR command).

TRACE:
0=no RFA trace is given.
1=RFA trace, for example the V2A set of protocol records are given. The FTP TRACE parameter is used for debugging. In normal use set it to 0.

DELETE:
0=the transferred subfile (if any) is not deleted at the local site if the transfer ends normally.
1=the transferred subfile (if any) is deleted at the local site if the transfer ends normally.

The UFT transferor will determine the restart point with the remote UFT server and the file transfer will restart from that point, avoiding unnecessary repetitive data transfers. The restart point is fixed by the receiver of the file (Requester or Remote Server).

NOTE:

Resources must be available for request execution. Since the execution is asynchronous, the DJP supervisor uses an automatic enqueuing/dequeuing mechanism to keep the request in a queue, until all the needed resources become available.

The request may be waiting for the communications servers to be started up, for the session connection to the remote site, or for the availability of local or remote files.

Structure of <infile_literal> and <outfile_literal>

```
01    FILE-LITERAL.
  02    FILE-LITERAL-LENGTH   COMP-1.
  02    FILE-LITERAL-STRING   PIC X(255).
```

The input and output file literals have the same structure with the only difference that one of the files is local and the other is a remote. Either file can be the infile or the outfile.

local_file_literal being the host dps7_file_literal of the format:

```
external_file_name [..subfile_name]
[:media:devclass] [$CATi] [$MFTj]
```

where:

external_file_name: mandatory

subfile_name: only for queued files

media:devclass: not specified if the file is resident or cataloged.

$CATi: i is a digit identifying the rank of the private Catalog file currently attached.

$MFTj: j is the FSN (file sequence number) for multi-file tapes.

remote_file_literal being the file on any remote site of the format:

```
      { dps7_file_literal    }
$site {                      }
      { foreign_file_literal }
```

where:

site: identifies the machine on which the remote file resides.

NOTE:

Problems may occur using the Remote File Literal, unless you are aware of the following information: MAIN station is often replaced by HOSTID station, for example the name of the DPS 7. If both the HOSTID station and the MAIN station exist in the CATALOG, HOSTID is always searched first in the catalog. This occurs every time MAIN is mentioned in the JCL/GCL commands. In this case, all sites must be attached to HOSTID. If MAIN only exists in the catalog, then attach all sites to MAIN. See the *DJP User's Guide* for more details.

dps7_file_literal: where the remote site is another DPS 7/7000.

foreign_file_literal: identifies a file located on a non-DPS 7/7000 system. The syntax of this file literal depends on the remote operating system. No syntactical analysis is performed on the file_literal_string.

Structure of SUBFT-EXECUTION-RESULT

```
01    SUBFT-EXECUTION-RESULT.
  02    SUBFT-RESULT          COMP-2.
  02    REQUEST-INDEX         COMP-2.
```

SUBFT-RESULT: identifies the result of the execution of the programmatic interface; see below for the list of values.

REQUEST-INDEX: is the file transfer internal index allocated by the UFT queue manager. This index is the FON (File Transfer Occurrence Number). If, for example, the value 1000 is returned to the caller, the generated request can be displayed:

- by the GCL command DUR (Display User Request): DUR F1000;
- or by a call to the procedure H_NP_UTESTFT,

subft_result	error diagnostic or wrong value for parameter
0	request submission is successful.
1	subft_length
2	request_priority
3	request_hold
4	request_class
5	binary
6	ncompact
7	append
8	restart
9	when
10	repeat
11	every
12	wait_file_transfer
13	If request submitter is given, password must be given, and vice versa
14	rst
15	Unable to retrieve TDS user/password
16	BRIEF parameter
17	TRACE parameter
20	input_file_literal
21	output_file_literal

22	local file is cataloged and the CATALOG file description cannot be retrieved
23	system Information about the remote SITE cannot be retrieved.
24	third party processing is not supported. Both infile and outfile are remote files. Launching a file transfer between two remote sites is not yet supported.
25	INDEF parameter/structure is erroneous.
26	OUTDEF parameter/structure is erroneous.
27	OUTALC parameter/structure is erroneous
30	The caller's parameters are syntactically correct but the request submission failed. The principal reasons may be: – the SITE specified in the remote_file_literal is unknown – or is not accessible to the project of the request submitter – or the parameter wait_file_transfer refers to unknown file transfer requests – or to system errors during the creation of the request.
40	system error: the request index cannot be allocated.

Structure of the TDS_COMMITMENT_IDENTIFICATION

In the TDS environment, TDS-COMMITMENT-IDENTIFICATION describes the commitment and the Request Submitter. Note that in V720, the password of the Request submitter has been added. If user is given, then a password must be given, and vice versa. The REQUEST-SUBMITTER-ID is used to connect to the remote site. It is not checked on the local site (the site submitting the CALL).

This structure has the following form:

```
+--------------------------------------------------+
|   01    TDS-COMMITMENT-IDENTIFICATION.           |
|    02   PROCESSOR-ID                 PIC X(4).  |
|    02   COMMIT-ID                    COMP-2.    |
|    02   TPR-ID                       COMP-1.    |
|    02   REQUEST-SUBMITTER-ID.                   |
|     03  USER                         PIC X(12).|
|     03  PROJECT                      PIC X(12).|
|     03  BILLING                      PIC X(12).|
|     03  PASSWORD                     PIC X(12).|
+--------------------------------------------------+
```

The TDS requests are "locked" (not selected for execution). Before the end of the commitment, TDS requests must be released or canceled using the H_NP_URELFT or H_NP_UCANFT system call.

In the COBOL environment, if INDEF and/or OUTDEF and/or OUTALC parameters need to be given, the TDS-COMMITMENT-IDENTIFICATION must exit and be set to spaces. This structure has the following form:

```
+--------------------------------------------------+
|   01  TDS-COMMITMENT-IDENTIFICATION.             |
|    02  TDS-COMMIT PIC X(58) VALUES SPACE.        |
+--------------------------------------------------+
```

If the file transfer is for someone else, USER, PROJECT, BILLING, and PASSWORD may be specified in the TDS-COMMITMENT-IDENTIFICATION. If the file transfer is launched from a COBOL batch program, PROCESSOR-ID must be set to SPACES, COMMIT-ID and TPR-ID must be set to 0, and the field PASSWORD of the COBOL-SUBFT-PARAMETERS must be set to “1”.

```
+----------------------------------------------------------+
|   01    TDS-COMMITMENT-IDENTIFICATION.                   |
|    02   PROCESSOR-ID           PIC X(4)   VALUE SPACES.  |
|    02   COMMIT-ID              COMP-2     VALUE 0.       |
|    02   TPR-ID                 COMP-1     VALUE 0.       |
|    02   REQUEST-SUBMITTER-ID.                            |
|     03  USER                   PIC X(12)  VALUE “U1”.    |
|     03  PROJECT                PIC X(12)  VALUE “P1”.    |
|     03  BILLING                PIC X(12)  VALUE “B1”.    |
|     03  PASSWORD               PIC X(12)  VALUE “PW”.    |
+----------------------------------------------------------+
```

Structure of the INDEF_PARAMETERS/OUTDEF_PARAMETERS

For file transfer with foreign machines such as DPS8 or DPX it is necessary to specify the record size, block size, and format. This information can be passed in structures using the DEF-DEFINE-SECTION. (see below).

Constraints are as follows:

- if INDEF-PARAMETERS and/or OUTDEF-PARAMETERS are required, then the TDS-COMMITMENT-IDENTIFICATION (set to spaces) is also required. (see above).
- if INDEF-PARAMETERS is not used but OUTDEF-PARAMETERS needs to be given, then the INDEF-PARAMETERS structure must exist, and the IFN must be set to spaces
- if OUTDEF-PARAMETERS is not used except the create option, then the OUTDEF-PARAMETERS structure must exist, and the IFN must be set to spaces. This allows the creation of an outfile as an image of the infile.

The define section must be declared as follows:

```
 01   DEF-DEFINE-SECTION.
*
* DEFINE header
*
   02 DEF-CREATE-OPTION PIC 1(8)  USAGE BIT.
      88 DEF-CREATE-OPT-NOT-SPECIFIED       VALUE B"00000000".
      88 DEF-CREATE-OPTION-KEEP             VALUE B"00000001".
      88 DEF-CREATE-OPTION-REPLACE          VALUE B"00000010".
      88 DEF-CREATE-OPTION-NEW              VALUE B"00000011".
   02 DEF-HEADER-RFU    PIC 1(24) USAGE BIT.
*
* COBOL equivalent of GPL
* $H-DEFINP NLVL1 NHEAD PREFIX=DEF- OVERLAY=CASS;
*
   02 DEF-IFN           PIC X(8).
   02 DEF-MASK          PIC 1(1) USAGE BIT.
      88 DEF-FUNCTION-MASK-SPECIFIED        VALUE B"1".
   02 DEF-TRUNCSSFF     PIC 1(1) USAGE BIT.
      88 DEF-TRUNCSSFF-SPECIFIED            VALUE B"1".
   02 DEF-NTRUNCSSFF    PIC 1(1) USAGE BIT.
      88 DEF-NTRUNCSSFF-SPECIFIED           VALUE B"1".
   02 DEF-BPIOC         PIC 1(1) USAGE BIT.
      88 DEF-BYPASS-IOCACHE-REQUESTED       VALUE B"1".
   02 DEF-DATACODE      PIC 1(4) USAGE BIT.
      88 DEF-DATACODE-NOT-SPECIFIED         VALUE B"0000".
      88 DEF-DATACODE-H200                  VALUE B"1000".
      88 DEF-DATACODE-BCD                   VALUE B"1001".
      88 DEF-DATACODE-EBCDIC                VALUE B"1100".
      88 DEF-DATACODE-ASCII                 VALUE B"1101".
   02 DEF-FILEFORM      PIC 1(8) USAGE BIT.
      88 DEF-FILEFORM-BFAS                  VALUE B"10000000".
      88 DEF-FILEFORM-OS360                 VALUE B"01000000".
      88 DEF-FILEFORM-DOS360                VALUE B"00100000".
      88 DEF-FILEFORM-HFAS                  VALUE B"00010000".
      88 DEF-FILEFORM-P6                    VALUE B"00001000".
      88 DEF-FILEFORM-UFAS                  VALUE B"00000100".
      88 DEF-FILEFORM-ANSI                  VALUE B"00000010".
      88 DEF-FILEFORM-NSTD                  VALUE B"00000001".
      88 DEF-FILEFORM-MLDS                  VALUE B"10000100".
      88 DEF-FILEFORM-SIRIS-3               VALUE B"00000011".
*                                    OTHERS VALUES RESERVED
   02 DEF-DUMMYREC      PIC 1(8) USAGE BIT.
*         FOR IND. SEQ.:DUMMY RECORD INSERTED EVERY N RECORDS
```

```
   02 DEF-RECFORM       PIC 1(8) USAGE BIT.
      88 DEF-RECFORM-NOT-SPECIFIED          VALUE B"00000000".
      88 DEF-RECFORM-F                      VALUE B"10000000".
      88 DEF-RECFORM-FB                     VALUE B"10010000".
      88 DEF-RECFORM-V                      VALUE B"01000000".
      88 DEF-RECFORM-VB                     VALUE B"01010000".
      88 DEF-RECFORM-U                      VALUE B"11000000".
      88 DEF-RECFORM-FS                     VALUE B"10001000".
      88 DEF-RECFORM-FBS                    VALUE B"10011000".
   02 DEF-RECSIZE       COMP-1.
*                              =0= NOT SPECIFIED
   02 DEF-BLOCKSZ       COMP-1.
*                              =0= NOT SPECIFIED
   02 DEF-COMSZ         PIC 1(8) USAGE BIT.
*              LENGTH OF COMMON SECTION IN CLI DEFINE RECORD
   02 DEF-BPB           PIC 1(8) USAGE BIT.
*                                NUMBER OF BLOCKS PER BUFFER
   02 DEF-CONV          PIC 1(2) USAGE BIT.
      88 DEF-CONV-CONV                      VALUE B"01".
      88 DEF-CONV-NCONV                     VALUE B"10".
   02 DEF-BSN           PIC 1(2) USAGE BIT.
      88 DEF-BSN-BSN                        VALUE B"01".
      88 DEF-BSN-NBSN                       VALUE B"10".
   02 DEF-JRNAL         PIC 1(2) USAGE BIT.
      88 DEF-JRNAL-BOTH                     VALUE B"11".
      88 DEF-JRNAL-BEFORE                   VALUE B"01".
      88 DEF-JRNAL-AFTER                    VALUE B"10".
      88 DEF-JRNAL-NONE                     VALUE B"00".
   02 DEF-JRNLF         PIC 1(1) USAGE BIT.
      88 DEF-JRNAL-SPECIFIED                VALUE B"1".
   02 DEF-NJRNAL        PIC 1(1) USAGE BIT.
      88 DEF-NBOTH-SPECIFIED                VALUE B"1".
      88 DEF-NBEFORE-SPECIFIED              VALUE B"1".
      88 DEF-NAFTER-SPECIFIED               VALUE B"1".
   02 DEF-PADCHARF      PIC 1(2) USAGE BIT.
      88 DEF-PADCHAR-SPECIFIED              VALUE B"00".
   02 DEF-PADCHAR       PIC 1(6) USAGE BIT.
*                                PADCHAR=OCTAL-2
   02 DEF-BANCHARF      PIC 1(2) USAGE BIT.
      88 DEF-BANCHAR-SPECIFIED              VALUE B"00".
   02 DEF-BANCHAR       PIC 1(6) USAGE BIT.
*                                BANCHAR=OCTAL-2
   02 DEF-KEYLOC        COMP-1.
*                                 KEY LOCATION
   02 DEF-LTRKSIZE      PIC 1(8) USAGE BIT.
*       =0= NOT SPECIFIED =1 TO 255= LOGICAL TRACK SIZE
   02 DEF-FUNCMASK      PIC 1(32) USAGE BIT.
*                                 FUNCMASK=HEXA-8
   02 DEF-SYSOUTF       PIC 1(1) USAGE BIT.
      88 DEF-SYSOUT-SPECIFIED               VALUE B"1".
   02 DEF-SYSOUT        PIC 1(1) USAGE BIT.
      88 DEF-SYSOUT-SYSOUT                 VALUE B"1".
      88 DEF-SYSOUT-NSYSOUT                VALUE B"0".
   02 DEF-WRCHECK       PIC 1(1) USAGE BIT.
      88 DEF-DISK-WRITE-CHECK              VALUE B"1".
   02 DEF-ERROPTF       PIC 1(1) USAGE BIT.
      88 DEF-ERROPT-SPECIFIED              VALUE B"1".
   02 DEF-ERROPT        PIC 1(2) USAGE BIT.
      88 DEF-ERROPT-RETCODE                VALUE B"00".
      88 DEF-ERROPT-SKIP                   VALUE B"01".
      88 DEF-ERROPT-ABORT                  VALUE B"10".
      88 DEF-ERROPT-IGNORE                 VALUE B"11".
   02 DEF-OPTIMIZE      PIC 1(1) USAGE BIT.
      88 DEF-OPTIMIZE-SPECIFIED            VALUE B"1".
      88 DEF-NO-OPTIMIZE                   VALUE B"0".
   02 DEF-MBZ-B1        PIC 1(1) USAGE BIT.
   02 DEF-ADDFORM       PIC 1(8) USAGE BIT.
      88 DEF-ADDFORM-TTRDD                 VALUE B"10000001".
      88 DEF-ADDFORM-LRRRR                 VALUE B"00100001".
      88 DEF-ADDFORM-SFRA                  VALUE B"00000010".
      88 DEF-ADDFORM-LRRR                  VALUE B"00100000".
   02 DEF-NDLRECF       PIC 1(1) USAGE BIT.
      88 DEF-NDLREC-SPECIFIED              VALUE B"1".
   02 DEF-DLRECF        PIC 1(1) USAGE BIT.
      88 DEF-DLREC-SPECIFIED               VALUE B"1".
   02 DEF-COMPACTF      PIC 1(1) USAGE BIT.
      88 DEF-COMPACT-SPECIFIED             VALUE B"1".
   02 DEF-NCOMPACTF     PIC 1(1) USAGE BIT.
      88 DEF-NCOMPACT-SPECIFIED            VALUE B"1".
   02 DEF-DVCODE        PIC 1(4) USAGE BIT.
      88 DEF-DVCODE-DISK                   VALUE B"0001".
      88 DEF-DVCODE-TAPE                   VALUE B"0010".
      88 DEF-DVCODE-READER-PUNCH           VALUE B"0011".
      88 DEF-DVCODE-PRINTER                VALUE B"0100".
      88 DEF-DVCODE-CASSETTE               VALUE B"1010".
   02 DEF-DATAFORMF     PIC 1(1) USAGE BIT.
      88 DEF-DATAFORM-SPECIFIED            VALUE B"1".
   02 DEF-DATAFORM      PIC 1(2) USAGE BIT.
      88 DEF-DATAFORM-SARF                 VALUE B"00".
      88 DEF-DATAFORM-SSF                  VALUE B"01".
      88 DEF-DATAFORM-DOF                  VALUE B"10".
      88 DEF-DATAFORM-ASA                  VALUE B"11".
   02 DEF-INKEYLOCF     PIC 1(1) USAGE BIT.
      88 DEF-INKEYLOC-SPECIFIED            VALUE B"1".
*                      VALUE OF INKEYLOC IS STORED IN KEYLOC
   02 DEF-BSILENF       PIC 1(1) USAGE BIT.
      88 DEF-BSILEN-SPECIFIED              VALUE B"1".
   02 DEF-CKPTLIMF      PIC 1(1) USAGE BIT.
      88 DEF-CKPTLIMF-SPECIFIED            VALUE B"1".
   02 DEF-KEYLOCF       PIC 1(1) USAGE BIT.
      88 DEF-KEYLOC-SPECIFIED              VALUE B"1".
   02 DEF-KEYSZF        PIC 1(1) USAGE BIT.
      88 DEF-KEYSIZE-SPECIFIED              VALUE B"1".
   02 DEF-BUFPOOL       PIC X(4).
*                BUFFER POOL IDENTIFICATION (DEFAULT=BLANK)
   02 DEF-CKPTLIM       COMP-2.
* CHECK POINT LIMIT:0=NO / NEGATIVE=EOV PLUS NUMBER / -1=EOV
   02 DEF-KEYSIZE       PIC 1(8) USAGE BIT.
*                               LENGTH OF KEY
   02 DEF-RFU-KEYLOC    PIC 1(8) USAGE BIT.
*                               RFU_KEYLOCATION
   02 DEF-CISIZE        COMP-1.
*                     NUMBER OF BYTES IN A CONTROL INTERVAL
   02 DEF-CASIZE        COMP-1.
*             NUMBER OF CONTROL INTERVALS IN A CONTROL AREA
   02 DEF-CIFSP         PIC 1(8) USAGE BIT.
*                               CONTROL INTERVAL FREE SPACE
   02 DEF-CAFSP         PIC 1(8) USAGE BIT.
*                               CONTROL AREA FREE SPACE
   02 DEF-CIFSPF        PIC 1(1) USAGE BIT.
      88 DEF-CIFSP-SPECIFIED                VALUE B"1".
   02 DEF-CAFSPF        PIC 1(1) USAGE BIT.
      88 DEF-CAFSP-SPECIFIED                VALUE B"1".
   02 DEF-FILELOADF     PIC 1(1) USAGE BIT.
      88 DEF-FILELOAD-SPECIFIED             VALUE B"1".
   02 DEF-FILELOAD      PIC 1(1) USAGE BIT.
      88 DEF-FILELOAD-ORDER                 VALUE B"0".
      88 DEF-FILELOAD-UNORDER               VALUE B"1".
   02 DEF-BYIDXF        PIC 1(1) USAGE BIT.
      88 DEF-BYIDX-SPECIFIED                VALUE B"1".
   02 DEF-BYIDX         PIC 1(1) USAGE BIT.
      88 DEF-BYIDX-NO                       VALUE B"0".
      88 DEF-BYIDX-YES                      VALUE B"1".
   02 DEF-IDXFSPF       PIC 1(1) USAGE BIT.
      88 DEF-IDXFSP-SPECIFIED               VALUE B"1".
   02 DEF-FORCE         PIC 1(1) USAGE BIT.
   02 DEF-BSILEN        PIC 1(8) USAGE BIT.
   02 DEF-DATABUF       COMP-1.
*                             DATA BUFFER NUMBER
   02 DEF-IDXFSP        PIC 1(8) USAGE BIT.
*                               IDXFSP VALUE
   02 DEF-READLOCKF     PIC 1(1) USAGE BIT.
      88 DEF-READLOCKF-SPECIFIED            VALUE B"1".
   02 DEF-READLOCK      PIC 1(2) USAGE BIT.
      88 DEF-READLOCK-NORMAL                VALUE B"00".
      88 DEF-READLOCK-EXCL                  VALUE B"01".
      88 DEF-READLOCK-STAT                  VALUE B"10".
   02 DEF-LOCKMARKF     PIC 1(1) USAGE BIT.
      88 DEF-LOCKMARK-SPECIFIED             VALUE B"1".
   02 DEF-FILEORGF      PIC 1(1) USAGE BIT.
      88 DEF-FILEORG-SPECIFIED              VALUE B"1".
   02 DEF-NBULBF        PIC 1(1) USAGE BIT.
      88 DEF-NBULB-SPECIFIED                VALUE B"1".
*                        NUMBER OF USER LABELS IS SPECIFIED
   02 DEF-RAHEAD        PIC 1(1) USAGE BIT.
      88 DEF-LMC-READ-AHEAD-REQUESTED       VALUE B"1".
   02 DEF-FRIOC         PIC 1(1) USAGE BIT.
      88 DEF-FORCE-IOCACHE-REQUESTED        VALUE B"1".
   02 DEF-FILEORG       PIC 1(8) USAGE BIT.
      88 DEF-FILEORG-NONE                   VALUE B"00000000".
      88 DEF-FILEORG-INDEXED                VALUE B"10000000".
      88 DEF-FILEORG-SEQUENTIAL             VALUE B"01000000".
      88 DEF-FILEORG-DIRECT                 VALUE B"00100000".
      88 DEF-FILEORG-RANDOM                 VALUE B"00010000".
      88 DEF-FILEORG-LINKED-QUEUED          VALUE B"00001000".
      88 DEF-FILEORG-MLDS                   VALUE B"00000100".
      88 DEF-FILEORG-RELATIVE-UFAS          VALUE B"00000010".
   02 DEF-NBULBL        PIC 1(8) USAGE BIT.
*                          NUMBER OF USER LABELS (0<= <=255)
   02 DEF-CASSETTE      PIC X(8).
*                                CASSETTE IDENTIFICATION
```

Structure of OUTALC-PARAMETERS

The 8th parameter enables the specification of the allocation characteristics of the output file to be created. If OUTALC is not used with the create option specified in OUTDEF, the output file will be created as the image of the input file.

The structure has the following form:

```
+---------------------------------------------+
|                                             |
|  01 OUTALC-PARAMETERS.                       |
|    02 OUTALC-SIZE       COMP-2.             |
|    02 OUTALC-INCRSIZE   COMP-1.             |
|    02 OUTALC-UNIT       PIC 1(8) USAGE BIT.|
|                                             |
+---------------------------------------------+
```

OUTALC-SIZE	specifies the amount of space required for the output file to be created.
OUTALC-INCRSIZE	specifies the automatic increment each time storage for output file is completely filled.
OUTALC-UNIT	specifies the unit of allocation for output file to be created: B"00000001" for RECORD B"00001000" for BLOCK B"00010000" for 100KB

Constraints are as follows:

- If OUTALC-PARAMETERS are set, then the TDS-COMMITMENT-IDENTFICATION must be set to spaces.
- If OUTALC-PARAMETERS are set, then the INDEF-PARAMETERS must be set, with the IFN set to spaces.
- If OUTALC-PARAMETERS are set, then the OUTDEF-PARAMETERS must be set with create option.
- If OUTALC-PARAMETERS are set, then the OUTALC-SIZE, OUTALC-INCRSIZE, OUTALC-UNIT parameters must be set.

9.3 Status Test Request Interface (H_NP_UTESTFT)

Programmatic Format

```
CALL "H_NP_UTESTFT" USING
      SUBFT-EXECUTION-RESULT,
      SUBFT-REQUEST-STATUS.
```

Structure of SUBFT-EXECUTION-RESULT

The following COBOL structure will receive values giving the Results of the Execution.

```
01    SUBFT-EXECUTION-RESULT.
  02    SUBFT-RESULT          COMP-2.
  02    REQUEST-INDEX         COMP-2.
```

SUBFT-RESULT *output parameter:* a number to identify the result of the execution of the test function of the programmatic interface

0 =	= the request has been found.
1 =	the request was not found which could imply that the given request index was wrong.
2 =	the request exists but the given index does not correspond to a File Transfer request.
3 =	several requests have the same number. This is normally not possible since it is the request index which is given in input and not a request name.
4 =	the request does not belong to the submitter of the CALL.
5 =	the request has been deleted by GTP for various reasons.
6 =	the request has been deleted by the user

NOTE:
The G4 general register status register is set as follows to:

- DONE if SUBFT-RESULT is 0
- OPTERR if SUBFT-RESULT is other than 0.

REQUEST-INDEX	*input parameter:* the file transfer internal index allocated by the DJP/UFT queue manager, whose status is being requested.

Structure of SUBFT-REQUEST-STATUS

The following COBOL structure will receive values giving the Status of the Request.

```
01    SUBFT-REQUEST-STATUS.
  02    REQUEST-STATUS        PIC X(4).
  02    TRANSFERRED-RECORDS   COMP-2.
```

REQUEST-STATUS

output parameter: the status of the request in 4 characters (if it exists and if it is a file transfer request).

This status can be:

CAN:
the request has been canceled by a GTP for various reasons such as EFN UNKNOWN or SUBFILE UNKNOWN.

DONE:
request found and completed, and the number of transferred records is given in the second field of the structure.

EX:
request is in execution and the number of transferred records at the time of TESTFT function is given in the second field of the structure.

HOLD:
the request was held at introduction time; issue RUR to release it.

RDY:
request is either waiting on a date/time condition (WDTM) or on an end of file transfer request (WTFR).

WAIT:
request ready for execution but not yet taken by a GTP.

WTFL:
request waiting on file availability; the dequeuing is automatic.

TRANSFERRED-RECORDS
output parameter: if request is EX, gives the number of transferred records at the time when H_NP_UTESTFT was executed.

❑

A. Copy Files For the Catalog OUTFILEs

This Appendix describes the COBOL structures of the records written to the output file of the LIST_CATALOG (LSCAT) and MAINTAIN_CATALOG (MNCAT) commands. This output file is specified via the OUTFILE parameter of these 2 commands.

A.1 The OUTFILE Parameter In Catalog Commands

The OUTFILE parameter enables you to place the output from the LIST_CATALOG (LSCAT) and MAINTAIN_CATALOG (MNCAT) commands in a file (for subsequent processing). The information placed in OUTFILE is basically the same as that written to PRTFILE. However, OUTFILE is written in a precisely defined structured format.

The syntax of the OUTFILE parameter is:

```
[ OUTFILE = output-file-description ]
```

where output-file-description is a standard JCL/GCL description of a sequential output file.

OUTFILE is made up of a sequence of SARF records of fixed size (175 bytes). If the file provided by the user has shorter records, then the information is truncated. The records are initialized to spaces before processing. Therefore, if a field is not applicable or empty, it remains filled with spaces.

The general structure of an OUTFILE record is:

```
* General record structure
* (From $H_DCOUTFREC PREFIX=GL_ ; REV: 01 (97.04.21))
 01  GL-REC.
  02  GL-ITEM.
   03  GL-CODE            PIC X(2).
   03  GL-VERSION         PIC X(2).
   03  GL-DATA            PIC X(171).
* End of General record structure
```

where:

GL-REC	General record structure.
GL-CODE	Code of the record type.
GL-VERSION	Version Number = "01".
GL-DATA	The rest of the record. This depends on the record type. It can contain text or numeric fields.

There are GPL macros, named H_DCOUTFxxxx which generate the GPL structures (when called with ’ATTRIB=cc_’ parameter, where cc is the code of the corresponding record, cc=HD, cc=RT, cc=VH, etc.). These macros also generate the equivalent COBOL structures cc-xxxx (with underscore (_) replaced by hyphen (-), CHAR(nn) replaced by PIC(nn), etc.).

A user program can use these structures via the COBOL statement:

```
COPY .... REPLACING LEADING cc- BY ...
```

The names of the files containing the COBOL structures are of the form: H-DCT-OUTF-xxxx. The list of file names is as follows:

```
H-DCT-OUTF-ACL
H-DCT-OUTF-ALLOC1
H-DCT-OUTF-ALLOC2
H-DCT-OUTF-ALLOC3
H-DCT-OUTF-APPL
H-DCT-OUTF-BILL
H-DCT-OUTF-COB
H-DCT-OUTF-COBX
H-DCT-OUTF-CONTROL
H-DCT-OUTF-ENVIRON
H-DCT-OUTF-FCLASS
H-DCT-OUTF-FLINK
H-DCT-OUTF-GEN
H-DCT-OUTF-HEAD
H-DCT-OUTF-LREC
H-DCT-OUTF-MEDIA
H-DCT-OUTF-MLINK
H-DCT-OUTF-NODE
H-DCT-OUTF-PATH
H-DCT-OUTF-PROJ
H-DCT-OUTF-REC
H-DCT-OUTF-ROOT
H-DCT-OUTF-SITE
H-DCT-OUTF-STAT
H-DCT-OUTF-STATION
H-DCT-OUTF-USER
H-DCT-OUTF-VOLHD
H-DCT-OUTF-VOLTAB
```

A.2 LSCAT Record Structures

The record structures described in paragraphs A.2.1 through A.2.8 apply to OUTFILEs generated by the LIST_CATALOG (LSCAT) command.

A.2.1 LREC Record

This structure is generated by the GPL macro H_DCOUTFLREC. It can be used each time a record is read to determine the type of record. Once the record type is determined, the appropriate specific record structure can be mapped onto it.

```
* General LSCAT record structure
* (From $H_DCOUTFLREC PREFIX=GL_ ; REV: 01 (97.05.13))
 01  GL-LREC.
  02  GL-ITEM.
   03  GL-CODE          PIC X(2).
   03  GL-VERSION       PIC X(2).
   03  GL-OBJ-NAME      PIC X(44).
   03  GL-OBJ-INVAL     PIC X(1).
   03  GL-OBJ-UNSTAB    PIC X(1).
   03  GL-OBJ-DATA      PIC X(125).
* End of General LSCAT structure
```

The field descriptions are as follows:

GL-LREC	General LSCAT record structure.
GL-CODE	Code of record type.
GL-VERSION	Version Number = "01".
GL-OBJ-NAME	Object Name.
GL-OBJ-INVAL	Invalid Object = "I".
GL-OBJ-UNSTAB	Unstable Object = "U".
GL-OBJ-DATA	The rest of the record.

A.2.2 HEAD Record

This is the structure of the Header record written to OUTFILE by the LIST_CATALOG (LSCAT) command. It is the first record written to OUTFILE

```
* Header Record
* (From $H_DCOUTFHEAD PREFIX=HD_ ; REV: 01 (97.05.13))
 01  HD-HEAD.
  02  HD-ITEM.
   03  HD-CODE          PIC X(2).
   03  HD-VERSION       PIC X(2).
   03  HD-OBJ-NAME      PIC X(44).
   03  FILLER           PIC X(2).
   03  HD-HD-MD         PIC X(6).
   03  HD-HD-DVC        PIC X(40).
   03  HD-HD-SMD        PIC X(6) OCCURS 10.
   03  FILLER           PIC X(19).
* End of Header Record
```

The field descriptions are as follows:

HD-HEAD	Header record structure.
HD-CODE	Code of record type = "HD".
HD-VERSION	Version Number = "01".
HD-OBJ-NAME	Catalog Name.
HD-HD-MD	Media containing the catalog (" " or Resident).
HD-HD-DVC	Selected device class (if any).
HD-HD-SMD	Selected media (if any).

A.2.3 ROOT Record

This is the structure of the record output at the root of the file system by the LIST_CATALOG (LSCAT) command. It corresponds to the PRTFILE record "*(root)".

```
* Root Record
  (From $H_DCOUTFROOT PREFIX=RT_ ; REV: 01 (97.05.13))
 01  RT-ROOT.
  02  RT-ITEM.
   03  RT-CODE          PIC X(2).
   03  RT-VERSION       PIC X(2).
   03  RT-OBJ-NAME      PIC X(44).
   03  RT-OBJ-INVAL     PIC X(1).
   03  FILLER           PIC X(1).
   03  RT-NB-OBJ        PIC X(6).
   03  RT-NB-BLK        PIC X(6).
   03  FILLER           PIC X(113).
* End of Root Record
```

The field descriptions are as follows:

RT-ROOT	Root record structure.
RT-CODE	Code of record type = "RT".
RT-VERSION	Version Number = "01".
RT-OBJ-NAME	Object Name.
RT-OBJ-INVAL	= "I", if the SITE.CATALOG is invalid.
RT-NB-OBJ	Number of objects.
RT-NB-BLK	Number of blocks.

A.2.4 VOLHD Record

This is the structure of the record output if the SORT-BY-VOLUME option was specified in the LIST_CATALOG (LSCAT) command. It is generated each time the next object is to be found on another volume.

```
* Volume-Header Record
*  (From $H_DCOUTFVOLHD PREFIX=VH_ ; REV: 01 (97.05.13))
 01  VH-VOLHD.
  02  VH-ITEM.
   03  VH-CODE          PIC X(2).
   03  VH-VERSION       PIC X(2).
   03  VH-OBJ-NAME      PIC X(44).
   03  FILLER           PIC X(2).
   03  VH-MD            PIC X(6).
   03  VH-DVC           PIC X(40).
   03  FILLER           PIC X(79).
* End of Volume-Header Record
```

The field descriptions are as follows:

VH-VOLHD	Volume record structure.
VH-CODE	Code of record type = "VH".
VH-VERSION	Version Number = "01".
VH-OBJ-NAME	Object Name.
VH-MD	Media containing the next object.
VH-DVC	Device class (of media).

A.2.5 Object Records

The description of any object begins with a record of this group. It may be preceded by a VOLHD record (see paragraph A.2.4). It may be followed by some Feature records (see paragraph A.2.6) which provide extra information. The Feature records depend on the object type and on the options specified by the user (when executing the LSCAT command).

A.2.5.1 NODE Record

This is the structure of the record output for directories by the LIST_CATALOG (LSCAT) command. A record is output for each directory (including the Master Directory). The Master Directory is distinguished by the absence of the dot (".") character in its name.

```
* Node (directory) Record
* (From $H_DCOUTFNODE PREFIX=ND_ ; REV: 01 (97.05.13))
 01  ND-NODE.
  02  ND-ITEM.
   03  ND-CODE          PIC X(2).
   03  ND-VERSION       PIC X(2).
   03  ND-OBJ-NAME      PIC X(44).
   03  ND-OBJ-INVAL     PIC X(1).
   03  ND-OBJ-UNSTAB    PIC X(1).
   03  FILLER           PIC X(125).
* End of Node (directory) Record
```

The field descriptions are as follows:

ND-NODE	Node record structure.
ND-CODE	Code of record type = "ND".
ND-VERSION	Version Number = "01".
ND-OBJ-NAME	Directory Name.
ND-OBJ-INVAL	= "I", if an Invalid Directory.
ND-OBJ-UNSTAB	= "U", if an Unstable Directory.

A.2.5.2 FCLASS Record

This is the structure of the record output for file generation groups by the LIST_CATALOG (LSCAT) command. A record is output for each file generation group, for the file generation, and for each ordinary file. The file generation group is distinguished by its subtype (FC-OBJ-SUBT = "G"). The file generation is distinguished from an ordinary file by its name. A file generation name contains "*G".

```
* File-Class (file) Record
*  (From $H_DCOUTFFCLASS PREFIX=FC_ ; REV: 01 (97.05.13))
 01  FC-FCLASS.
  02  FC-ITEM.
   03  FC-CODE          PIC X(2).
   03  FC-VERSION       PIC X(2).
   03  FC-OBJ-NAME      PIC X(44).
   03  FC-OBJ-INVAL     PIC X(1).
   03  FC-OBJ-UNSTAB    PIC X(1).
   03  FC-OBJ-SUBT      PIC X(1).
   03  FILLER           PIC X(124).
* End of File-Class (file) Record
```

The field descriptions are as follows:

FC-FCLASS	File Class record structure.
FC-CODE	Code of record type = "FC".
FC-VERSION	Version Number = "01".
FC-OBJ-NAME	Object Name.
FC-OBJ-INVAL	= "I", in an Invalid Object.
FC-OBJ-UNSTAB	= "U", if an Unstable Object.
FC-OBJ-SUBT	= "G" if a Generation Group, = " " if a file or a file generation.

A.2.5.3 FLINK Record

This is the structure of the record output for file links by the LIST_CATALOG (LSCAT) command. A record is output for each file link.

```
* File-Link Record
*  (From $H_DCOUTFFLINK PREFIX=FL_ ; REV: 01 (97.05.13))
 01  FL-FLINK.
  02  FL-ITEM.
   03  FL-CODE             PIC X(2).
   03  FL-VERSION          PIC X(2).
   03  FL-OBJ-NAME         PIC X(44).
   03  FL-OBJ-INVAL        PIC X(1).
   03  FL-OBJ-UNSTAB       PIC X(1).
   03  FL-PATH-TYPE        PIC X(1).
   03  FL-PATH-NAME        PIC X(44).
   03  FILLER              PIC X(80).
* End of File-Link Record
```

The field descriptions are as follows:

FL-FLINK	File Link record structure.
FL-CODE	Code of record type = "FL".
FL-VERSION	Version Number = "01".
FL-OBJ-NAME	Object Name.
FL-OBJ-INVAL	= "I", if an Invalid Object.
FL-OBJ-UNSTAB	= "U", if an Unstable Object.
FL-PATH-TYPE	= "M", if an MLDSPATH.
FL-PATH-NAME	Name of the first path.

A.2.5.4 MLINK Record

This is the structure of the record output for master links by the LIST_CATALOG (LSCAT) command. A record is output for each master link.

```
* Master-Link Record
*  (From $H_DCOUTFMLINK PREFIX=ML_ ; REV: 01 (97.05.13))
 01  ML-MLINK.
  02  ML-ITEM.
   03  ML-CODE            PIC X(2).
   03  ML-VERSION         PIC X(2).
   03  ML-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  ML-ML-STRUCT       OCCURS 6.
    04  ML-ML-INVAL       PIC X(1).
    04  ML-ML-NAME        PIC X(16).
   03  FILLER             PIC X(23).
* End of Master-Link Record
```

The field descriptions are as follows:

ML-MLINK	Master Link record structure.
ML-CODE	Code of record type = "ML".
ML-VERSION	Version Number = "01".
ML-OBJ-NAME	Object Name.
ML-ML-INVAL	= "I", if an Invalid state.
ML-ML-NAME	Name of the Master Link.

A.2.6 Feature Records

A.2.6.1 CONTROL Record

This is the structure of the CONTROL record output by the LIST_CATALOG (LSCAT) command.

```
* Control Record
*  (From $H_DCOUTFCONTROL PREFIX=CT_ ; REV: 01 (97.05.13))
 01  CT-CONTROL.
  02  CT-ITEM.
   03  CT-CODE          PIC X(2).
   03  CT-VERSION       PIC X(2).
   03  CT-OBJ-NAME      PIC X(44).
   03  FILLER           PIC X(2).
   03  CT-GENTYPE       PIC X(1).
   03  CT-NBGEN1        PIC X(3).
   03  CT-NBGEN2        PIC X(4).
   03  CT-RETPER        PIC X(3).
   03  CT-IGEXPDT       PIC X(1).
   03  CT-SHARE         PIC X(4).
   03  CT-DUALSHR       PIC X(4).
   03  CT-JOURNAL       PIC X(4).
   03  CT-SYSTEM        PIC X(1).
   03  CT-LASTMDDATE    PIC X(17).
   03  CT-AUTOATT       PIC X(1).
   03  FILLER           PIC X(82).
* End of Control Record
```

The field descriptions are as follows:

CT-CONTROL	Control record structure.
CT-CODE	Code of record type = "CT".
CT-VERSION	Version Number = "01".
CT-OBJ-NAME	Object Name.
CT-GENTYPE	= " " or "O", if an Open loop generation, = "C", if a Closed loop generation.
CT-NBGEN1	Number of generations.

CT-NBGEN2	Generation number.
CT-RETPER	Duration of retention period.
CT-IGEXPDT	= "Y", if Ignore expiry date.
CT-SHARE	Share Option: "NORM", "ONEW", "MONI", "FREE", "DIR ", "UNSP", or "UNKN".
CT-DUALSHR	Dual Share Option: "NORM", "NONE", "ONEW", or "FREE".
CT-JOURNAL	Journal Option: "NO ", "BEFO", "AFTE", "BOTH", or "PRIV".
CT-SYSTEM	= "S", if System.
CT-LASTMDDATE	Last Modification Date: "YY.MM.DD/hh.mm.ss".
CT-AUTOATT	Auto-attachable: "N", "Y", or " ".

A.2.6.2 ACL Record

This is the structure of the ACL record output by the LIST_CATALOG (LSCAT) command.

```
* ACL Record
*  (From $H_DCOUTFACL PREFIX=AC_ ; REV: 01 (97.03.20))
 01  AC-ACL.
  02  AC-ITEM.
   03  AC-CODE            PIC X(2).
   03  AC-VERSION         PIC X(2).
   03  AC-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  AC-ACLRIGHT        PIC X(8).
   03  AC-ACLPROJECT      PIC X(12) OCCURS 8.
   03  FILLER             PIC X(21).
* End of ACL Record
```

The field descriptions are as follows:

AC-ACL	ACL record structure.
AC-CODE	Code of record type = "AC".
AC-VERSION	Version Number = "01".

AC-OBJ-NAME	Object Name.
AC-ACLRIGHT	ACL right.
AC-ACLPROJECT	Projects having the right (given by AC-ACLRIGHT) or " ".

A.2.6.3 GEN Record

This is the structure of the GEN record output by the LIST_CATALOG (LSCAT) command.

```
* GEN Record
*  (From $H_DCOUTFGEN PREFIX=GN_ ; REV: 01 (97.05.13))
 01  GN-GEN.
  02  GN-ITEM.
   03  GN-CODE            PIC X(2).
   03  GN-VERSION         PIC X(2).
   03  GN-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  GN-GENREL          PIC X(5).
   03  GN-GENSYMB         PIC X(5).
   03  FILLER             PIC X(115).
* End of GEN Record
```

The field descriptions are as follows:

GN-GEN	GEN record structure.
GN-CODE	Code of record type = "GN".
GN-VERSION	Version Number = "01".
GN-OBJ-NAME	Object Name.
GN-GENREL	Relative generation number.
GN-GENSYMB	Symbolic generation name.

A.2.6.4 PATH Record

This is the structure of the PATH record output by the LIST_CATALOG (LSCAT) command.

```
* Path Record
*  (From $H_DCOUTFPATH PREFIX=PT_ ; REV: 01 (97.05.13))
 01  PT-PATH.
  02  PT-ITEM.
   03  PT-CODE            PIC X(2).
   03  PT-VERSION         PIC X(2).
   03  PT-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  PT-PATH-NAMES      PIC X(44)  OCCURS 2.
   03  FILLER             PIC X(37).
* End of Path Record
```

The field descriptions are as follows:

PT-PATH	PATH record structure.
PT-CODE	Code of record type = "PT".
PT-VERSION	Version Number = "01".
PT-OBJ-NAME	Object Name.
PT-PATH-NAMES	Path names.

A.2.7 ALLOC Records

These records are: ALLOC-1, ALLOC-2, MEDIA, and ALLOC-3.

A.2.7.1 ALLOC-1

This is the structure of the ALLOC-1 record output by the LIST_CATALOG (LSCAT) command.

```
* Alloc-1 Record
*  (From $H_DCOUTFALLOC1 PREFIX=A1_ ; REV: 01 (97.05.13))
 01  A1-ALLOC1.
  02  A1-ITEM.
   03  A1-CODE            PIC X(2).
   03  A1-VERSION         PIC X(2).
   03  A1-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  A1-IOC             PIC X(1).
   03  A1-LOGSUBF         PIC X(1).
   03  A1-MIGRATED        PIC X(4).
   03  A1-SAVED-BY        PIC X(4).
   03  A1-RES-FACT        PIC X(6).
   03  A1-LASTREF         PIC X(17).
   03  A1-LASTSAVE        PIC X(17).
   03  A1-LASTUPDATE      PIC X(17).
   03  A1-CREATED         PIC X(17).
   03  A1-UNIT            PIC X(3).
   03  A1-SIZE            PIC X(10).
   03  A1-INCRSIZE        PIC X(6).
   03  A1-INCRUNIT        PIC X(3).
   03  A1-ON-RESID        PIC X(1).
   03  FILLER             PIC X(18).
* End of Alloc-1 Record
```

The field descriptions are as follows:

A1-ALLOC1	ALLOC-1 record structure.
A1-CODE	Code of record type = "A1".
A1-VERSION	Version Number = "01".

A1-OBJ-NAME	Object Name.
A1-IOC	I/O Cache State: "D" = Default, "F" = Force, "B" = Bypass, "U" = Unspecified.
A1-LOGSUBF	Library modifications logged: "Y" or "N".
A1-MIGRATED	Migrated by (with de-allocation).
A1-SAVED-BY	Pre-migrated by (without de-allocation)
A1-RES-FACT	Residency factor.
A1-LASTREF	Last reference: "YY.MM.DD/hh.mm.ss".
A1-LASTSAVE	Last save: "YY.MM.DD/hh.mm.[ss]".
A1-LASTUPDATE	Last update: "YY.MM.DD/hh.mm.[ss]".
A1-CREATED	Created: "YY.MM.DD/hh.mm.ss".
A1-UNIT	Allocation unit: "CYL", etc.
A1-SIZE	Size: in allocation units (given by A1-UNIT).
A1-INCRSIZE	Increment size.
A1-INCRUNIT	Unit of increment size.
A1-ON-RESID	= "Y", if a System file on a Resident Disk.

A.2.7.2 ALLOC-2

This is the structure of the ALLOC-2 record output by the LIST_CATALOG (LSCAT) command.

```
* Alloc-2 Record
*  (From $H_DCOUTFALLOC2 PREFIX=A2_ ; REV: 01 (97.05.13))
 01  A2-ALLOC2.
  02  A2-ITEM.
   03  A2-CODE             PIC X(2).
   03  A2-VERSION          PIC X(2).
   03  A2-OBJ-NAME         PIC X(44).
   03  FILLER              PIC X(2).
   03  A2-SLOCK            PIC X(3).
   03  A2-PMD              PIC X(2).
   03  A2-ABT-LOCK         PIC X(3).
   03  FILLER              PIC X(116).
* End of Alloc-2 Record
```

The field descriptions are as follows:

A2-ALLOC2	ALLOC-2 record structure.
A2-CODE	Code of record type = "A2".
A2-VERSION	Version Number = "01".
A2-OBJ-NAME	Object Name.
A2-SLOCK	Security Lock: "OFF" or "ON ".
A2-PMD	Processing Mode: "IN", "AP", "IA", or " ".
A2-ABT-LOCK	Abort Lock: "OFF" or "ON ".

A.2.7.3 MEDIA

This is the structure of the MEDIA record output by the LIST_CATALOG (LSCAT) command.

```
* Media Record
*  (From $H_DCOUTFMEDIA PREFIX=MD_ ; REV: 01 (97.05.13))
 01  MD-MEDIA.
  02  MD-ITEM.
   03  MD-CODE            PIC X(2).
   03  MD-VERSION         PIC X(2).
   03  MD-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  MD-MEDIAT      PIC X(6)  OCCURS 20.
   03  FILLER             PIC X(5).
* End of Media Record
```

The field descriptions are as follows:

MD-MEDIA	MEDIA record structure.
MD-CODE	Code of record type = "MD".
MD-VERSION	Version Number = "01".
MD-OBJ-NAME	Object Name.
MD-MEDIAT	Media Table.

A.2.7.4 ALLOC-3

This is the structure of the ALLOC-3 record output by the LIST_CATALOG (LSCAT) command.

```
* Alloc-3 Record
*  (From $H_DCOUTFALLOC3 PREFIX=A3_ ; REV: 01 (97.05.13))
 01  A3-ALLOC3.
  02  A3-ITEM.
   03  A3-CODE            PIC X(2).
   03  A3-VERSION         PIC X(2).
   03  A3-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  A3-VOLSET          PIC X(6).
   03  A3-PROTECT         PIC X(1).
   03  A3-DEVCLASS        PIC X(40).
   03  A3-NBVOLUSED       PIC X(2).
   03  FILLER             PIC X(76).
* End of Alloc-3 Record
```

The field descriptions are as follows:

A3-ALLOC3	ALLOC-3 record structure.
A3-CODE	Code of record type = "A3".
A3-VERSION	Version Number = "01".
A3-OBJ-NAME	Object Name.
A3-VOLSET	Name of VOLSET allocation.
A3-PROTECT	Protected by catalog: "Y", "N", or " ".
A3-DEVCLASS	Expanded device class.
A3-NBVOLUSED	Number of volumes used.

A.2.7.5 COBOL Text Record

COB

This is the structure of the COB record output by the LIST_CATALOG (LSCAT) command.

```
* Extract from COBOL Text-Record
*  (From $H_DCOUTFCOB PREFIX=CB_ ; REV: 01 (98.03.02))
 01  CB-REC.
  02  CB-ITEM.
   03  CB-CODE            PIC XX.
   03  CB-VERSION         PIC XX.
   03  CB-OBJ-NAME        PIC X(44).
   03  CB-ORG             PIC X(10).
   03  CB-CODESET         PIC X(10).
   03  CB-COLLSEQ         PIC X(10).
   03  CB-PAD             PIC X.
   03  CB-MINRECSIZE      PIC X(6).
   03  CB-MAXRECSIZE      PIC X(6).
   03  CB-BLOCK-SIZE      PIC X(6).
   03  CB-FILE-POS        PIC X(6).
   03  CB-RECFORM         PIC X.
   03  CB-BLOCKED         PIC X.
   03  CB-PADDING         PIC X.
   03  CB-DELIM           PIC X.
   03  CB-COMP-REC        PIC X.
   03  CB-SECIDX          PIC X.
   03  CB-W-SSF           PIC X.
   03  CB-W-SARF          PIC X.
   03  CB-W-ASA           PIC X.
   03  CB-W-BSN           PIC X.
   03  CB-NO-COMP-REC     PIC X.
   03  CB-PADDING-ON      PIC X.
   03  CB-PKSZ            PIC X(6).
   03  CB-PKLOC           PIC X(6).
   03  CB-ALT-KEY-NB      PIC X(6).
   03  CB-PROG-ID         PIC X(15).
   03  CB-CRDATE          PIC X(17).
   03  FILLER             PIC X(10).
* End of Extract from COBOL Text-Record
```

The field descriptions are as follows:

CB-REC	CB record structure (from COBOL created text-record).
CB-CODE	Code of record type = "CB".
CB-VERSION	Version Number = "01".
CB-OBJ-NAME	Object Name.
CB-ORG	ORGANIZATION clause.
CB-CODESET	CHARACTER CODESET clause.
CB-COLLSEQ	COLLATING SEQUENCE clause.
CB-PAD	PADDING character value.
CB-MINRECSIZE	Minimum record size.
CB-MAXRECSIZE	Maximum record size.
CB-BLOCK-SIZE	BLOCK CONTAINS clause value.
CB-FILE-POS	MULTIPLE FILE clause.
CB-RECFORM	"F" = fixed length records, "V" = variable length records.
CB-BLOCKED	BLOCK CONTAINS clause "Y".
CB-PADDING	PADDING clause = "Y", NO PADDING clause = "N".
CB-DELIM	RECORD DELIMITER clause: "Y" = STANDARD, "N" = IMPLIED.
CB-COMP-REC	Complementary records = "Y". This is an obsolete MLDS feature.
CB-SECIDX	Secondary Index = "Y". This is an obsolete MLDS feature.
CB-W-SSF	WITH SSF clause = "Y".
CB-W-SARF	WITH SARF clause = "Y".
CB-W-ASA	WITH ASA clause = "Y".
CB-W-BSN	WITH BSN clause = "Y".
CB-NO-COMP-REC	No complementary records = "Y". This is an obsolete MLDS feature.

CB-PADDING-DN	Data-name specified in PADDING CHARACTER clause.
CB-PKSZ	Primary key size.
CB-PKLOC	Primary key location (from 0).
CB-ALT-KEY-NB	Number of alternate keys.
CB-PROG-ID	Name of COBOL program that created the text-record.
CB-CRDATE	Date and time of the file catalog entry creation.

COBX

This is the structure of the COBX record output by the LIST_CATALOG (LSCAT) command when alternate key(s) is/are described in the COBOL created text-record.

```
* Extract from COBOL Text-Record
*  (From $H_DCOUTFCOBX PREFIX=CX_ ; REV: 01 (98.03.02))
 01  CX-REC.
  02  CX-ITEM.
   03  CX-CODE            PIC XX.
   03  CX-VERSION         PIC XX.
   03  CX-OBJ-NAME        PIC X(44).
   03  CX-ALT_KEY_DESC    OCCURS 14.
      04  CX-ALT_KEY_SIZE PIC XXXX.
      04  CX-ALT_KEY_LOC  PIC XXXXXX.
      04  CX-ALT_KEY_DUPL PIC X.
   03  FILLER             PIC X.
* End of Extract from COBOL Text-Record
```

The field descriptions are as follows:

CX-REC	CX record structure (from text-record with alternate key(s)).
CX-CODE	Code of record type = "CX".
CX-VERSION	Version Number = "01".
CX-OBJ-NAME	Object Name.
CX-ALT_KEY_DESC	ALTERNATE KEY description.
CX-ALT_KEY_SIZE	ALTERNATE KEY size.
CX-ALT_KEY_LOC	ALTERNATE KEY location (from 0).
CX-ALT_KEY_DUP	ALTERNATE KEY WITH DUPLICATES: "Y" or "N".

A.2.8 Statistics Record

This is the structure of the STAT record output by the LIST_CATALOG (LSCAT) command. The STAT record is generated only if there are 2 or more objects listed. If this record is generated, it is the last record of OUTFILE.

```
* Statistics Record
*  (From $H_DCOUTFSTAT PREFIX=SS_ ; REV: 01 (97.05.13))
 01  SS-STAT.
  02  SS-ITEM.
   03  SS-CODE            PIC X(2).
   03  SS-VERSION         PIC X(2).
   03  SS-OBJ-NAME        PIC X(44).
   03  FILLER             PIC X(2).
   03  SS-NB-OF-OBJ       PIC X(5).
   03  SS-FREE-SPACE      PIC X(5).
   03  FILLER             PIC X(115).
* End of Statistics Record
```

The field descriptions are as follows:

SS-STAT	STAT record structure.
SS-CODE	Code of record type = "SS".
SS-VERSION	Version Number = "01".
SS-OBJ-NAME	Object Name (but not relevant in this case).
SS-NB-OF-OBJ	Number of objects listed.
SS-FREE-SPACE	Percentage of free space.

A.3 MNCAT Record Structures

The record structures described in paragraphs A.3.1 through A.3.7 apply to OUTFILEs generated by the MAINTAIN_CATALOG (MNCAT) command.

A.3.1 PROJ Record

This is the structure of the PROJ record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Project Record
*  (From $H_DCOUTFPROJ PREFIX=PR_ ; REV: 01 (97.04.21))
 01  PR-PROJ.
  02  PR-ITEM.
   03  PR-CODE           PIC X(2).
   03  PR-VERSION        PIC X(2).
   03  PR-LEVEL          PIC X(2).
   03  PR-PROJNAME       PIC X(12).
   03  PR-PROJISDFLT     PIC X(1).
   03  PR-MODIFDATE      PIC X(8).
   03  PR-JOBDFLT        PIC X(2).
   03  PR-JOBIOF         PIC X(2).
   03  PR-JOBCLASS       PIC X(2)  OCCURS 26.
   03  PR-DFLTOUTCLASS   PIC X(2).
   03  PR-PROJATTR       PIC X(8)  OCCURS 4.
   03  PR-PROJSTUP       PIC X(8)  OCCURS 4.
   03  FILLER            PIC X(26).
* End of Project Record
```

The field descriptions are as follows:

PR-PROJ	PROJ record structure.
PR-CODE	Code of record type = "PR".
PR-VERSION	Version Number = "01".
PR-LEVEL	Object level.
PR-PROJNAME	Project name.
PR-PROJISDFLT	= "Y", if it is a default project.
PR-MODIFDATE	Modification date: "YY.MM.DD".
PR-JOBDFLT	Batch default class.

PR-JOBIOF	IOF default class.
PR-JOBCLASS	Job class list.
PR-DFLTOUTCLASS	Default OUTPUT class.
PR-PROJATTR	Attributes: "STD ", "MAIN ", "STATION ", "RMS ", or " ".
PR-PROJSTUP	Startup: "SITE ", "PROJECT ", "USER ", or "EMPTY ".

A.3.2 VOLTAB Record

This is the structure of the VOLTAB record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Volumes table of project Record
*  (From $H_DCOUTFVOLTAB PREFIX=VT_ ; REV: 01 (97.04.21))
 01  VT-VOLTAB.
  02  VT-ITEM.
   03  VT-CODE           PIC X(2).
   03  VT-VERSION        PIC X(2).
   03  FILLER            PIC X(2).
   03  VT-VOLMT          PIC X(1).
   03  VT-VOLDESCR       OCCURS 20.
    04  VT-VSN           PIC X(6).
    04  VT-VSNPROTECT    PIC X(1).
    04  VT-VSNRANGE      PIC X(1).
   03  FILLER            PIC X(8).
* End of Volumes table of project Record
```

The field descriptions are as follows:

VT-VOLTAB	VOLTAB record structure.
VT-CODE	Code of record type = "VT".
VT-VERSION	Version Number = "01".
VT-VOLMT	= "Y", if volumes on MT.

VT-VSN	VSN.
VT-VSNPROTECT	= "Y", if VSN protected.
VT-VSNRANGE	= " ", if volume is a single volume, = "B", if volume is the beginning of a range of volumes, = "E", if volume is the end of a range of volumes.

A.3.3 APPL Record

This is the structure of the application record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Application Record
*  (From $H_DCOUTFAPPL PREFIX=AP_ ; REV: 01 (97.04.21))
 01  AP-APPL.
  02  AP-ITEM.
   03  AP-CODE           PIC X(2).
   03  AP-VERSION        PIC X(2).
   03  FILLER            PIC X(2).
   03  AP-APPLNAME       PIC X(12).
   03  AP-APPLISDFLT     PIC X(1).
   03  AP-APPLTDS        PIC X(8).
   03  FILLER            PIC X(148).
* End of Application Record
```

The field descriptions are as follows:

AP-APPL	Application record structure.
AP-CODE	Code of record type = "AP".
AP-VERSION	Version Number = "01".
AP-APPLNAME	Application name.
AP-APPLISDFLT	= "Y", if it is a default application.
AP-APPLTDS	TDS code.

A.3.4 ENVIRON Record

This is the structure of the environment record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Environment Record
*  (From $H_DCOUTFENVIRON PREFIX=EN_ ; REV: 01 (97.04.21))
 01  EN-ENVIRON.
  02  EN-ITEM.
   03  EN-CODE            PIC X(2).
   03  EN-VERSION         PIC X(2).
   03  FILLER             PIC X(2).
   03  EN-ENVTNAME        PIC X(12).
   03  EN-ENVTISDFLT      PIC X(1).
   03  FILLER             PIC X(156).
* End of Environment Record
```

The field descriptions are as follows:

EN-ENVIRON	Environment record structure.
EN-CODE	Code of record type = "EN".
EN-VERSION	Version Number = "01".
EN-ENVTNAME	Environment name.
EN-ENVTISDFLT	= "Y", if it is a default environment.

A.3.5 STATION Record

This is the structure of the station record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Station Record
*  (From $H_DCOUTFSTATION PREFIX=SN_ ; REV: 01 (97.04.21))
 01  SN-STATION.
  02  SN-ITEM.
   03  SN-CODE            PIC X(2).
   03  SN-VERSION         PIC X(2).
   03  SN-LEVEL           PIC X(2).
   03  SN-STTNNAME        PIC X(12).
   03  SN-STTNISDFLT      PIC X(1).
   03  SN-STTNTDS         PIC X(8).
   03  FILLER             PIC X(148).
* End of Station Record
```

The field descriptions are as follows:

SN-STATION	Station record structure.
SN-CODE	Code of record type = "SN".
SN-VERSION	Version Number = "01".
SN-LEVEL	Object level.
SN-STTNNAME	Station name.
SN-STTNISDFLT	"Y" = if it is a default station.
SN-STTNTDS	TDS code.

A.3.6 USER Record

This is the structure of the user record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* User Record
*  (From $H_DCOUTFUSER PREFIX=US_ ; REV: 01 (97.04.21))
 01  US-USER.
  02  US-ITEM.
   03  US-CODE           PIC X(2).
   03  US-VERSION        PIC X(2).
   03  US-LEVEL          PIC X(2).
   03  US-USRNAME        PIC X(12).
   03  US-USRISDFLT      PIC X(1).
   03  US-USRTDS         PIC X(8).
   03  US-USRMODIFDATE   PIC X(8).
   03  FILLER            PIC X(140).
* End of User Record
```

The field descriptions are as follows:

US-USER	User record structure.
US-CODE	Code of record type = "US".
US-VERSION	Version Number = "01".
US-LEVEL	Object level.
US-USRNAME	User name.
US-USRISDFLT	"Y" = a default user, " " not a default user.
US-USRTDS	TDS code.
US-USRMODIFDATE	Modification date: "YY.MM.DD".

A.3.7 BILL Record

This is the structure of the billing record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Billing Record
*  (From $H_DCOUTFBILL PREFIX=BI_ ; REV: 01 (97.04.21))
 01  BI-BILL.
  02  BI-ITEM.
   03  BI-CODE            PIC X(2).
   03  BI-VERSION         PIC X(2).
   03  BI-LEVEL           PIC X(2).
   03  BI-BILLNAME        PIC X(12).
   03  BI-BILLISDFLT      PIC X(1).
   03  BI-BILLMODIFDATE  PIC X(8).
   03  BI-BILLCREDIT      PIC X(11).
   03  BI-BILLCHARGE      PIC X(11).
   03  BI-BILLBALANCE     PIC X(11).
   03  FILLER             PIC X(115).
* End of Billing Record
```

The field descriptions are as follows:

BI-BILL	Billing record structure.
BI-CODE	Code of record type = "BI".
BI-VERSION	Version Number = "01".
BI-LEVEL	Object level.
BI-BILLNAME	Billing name.
BI-BILLISDFLT	"Y" = if it is a default billing.
BI-BILLMODIFDATE	Modification date: "YY.MM.DD".
BI-BILLCREDIT	Credit.
BI-BILLCHARGE	Charge.
BI-BILLBALANCE	Balance.

A.3.8 SITE Record

This is the structure of the site record output by the MAINTAIN_CATALOG (MNCAT) command.

```
* Site Record
*  (From $H_DCOUTFSITE PREFIX=SI_ ; REV: 01 (97.04.21))
 01  SI-SITE.
  02  SI-ITEM.
   03  SI-CODE           PIC X(2).
   03  SI-VERSION        PIC X(2).
   03  SI-LEVEL          PIC X(2).
   03  SI-SITENAME       PIC X(8).
   03  FILLER            PIC X(161).
* End of Site Record
```

The field descriptions are as follows:

SI-SITE	Site record structure.
SI-CODE	Code of record type = "SI".
SI-VERSION	Version Number = "01".
SI-LEVEL	Object level.
SI-SITENAME	Site name.

B. Copy files for LIST_VOLUME or LIST_FILE OUTFILE contents

This appendix describes the COBOL structures of the records written to the output file of the LIST_VOLUME (LSV) or LIST_FILE (LSF) command. Besides the records described hereunder, records of Catalog OUTFILES, described in appendix A (especially ACL, CONTROL and COBOL) can also appear.

The output file is specified via the OUTFILE parameter of the command.

B.1 The OUTFILE parameter in LIST_VOLUME or LIST_FILE command

The OUTFILE parameter enables you to place the output from the LIST_VOLUME (LSV) or LIST_FILE (LSF) command in a file (for subsequent processing). The information placed in OUTFILE is basically the same as that written to PRTFILE. However, OUTFILE is written in a precisely defined structured format.

The syntax of the OUTFILE parameter is :

[OUTFILE = output-file-description]

where output-file-description is a standard JCL/GCL description of sequential output file.

OUTFILE is made up of a sequence of SARF records of fixed size (175 bytes). If the file provided by the user has shorter records, then the information is truncated. The records are initialized to spaces before processing. Therefore, if a field is not applicable or empty, it remains filled with spaces.

The general structure of an OUTFILE record is :

```
01 CB-FVL-REC.
    05 CB-CODE          PIC X(2).
    05 CB-VERSION       PIC X(2).
    05 CB-DATA          PIC X(171).
```

Where CB-CODE is the record type. CB-DATA depends on the record type, it can contain text or numeric fields.

The “US” record contains bit strings ; for the corresponding COPY, H-DCT-FVL-USAUNS, NSTD level of Cobol is necessary.

The names of the COBOL COPYs containing the OUTFILE record structures are of the form H-DCT-FVL-xxx. There are GPL macros, named H_DCFVLxxx which generate the equivalent GPL structures.

A user program can use these COPYs via the COBOL statement :

```
COPY H-DCT-FVL-xxx REPLACING LEADING CB- BY …
```

The list of these COPY names is as follows, with the corresponding code (in code alphabetical order) :

H-DCT-FVL-USAASI	AS
H-DCT-FVL-DESC	DE
H-DCT-FVL-ERROR	ER
H-DCT-FVL-USAKEY	KY
H-DCT-FVL-USALIB	LI
H-DCT-FVL-ORG1	O1
H-DCT-FVL-ORG2	O2
H-DCT-FVL-SAVINFO	SI
H-DCT-FVL-SPACE	SP
H-DCT-FVL-USASTAT	ST
H-DCT-FVL-SIZE	SZ
H-DCT-FVL-TITRE	TI
H-DCT-FVL-USAUFAS	UF
H-DCT-FVL-USAUNS	US
H-DCT-FVL-USA1	U1
H-DCT-FVL-VOLUME	V1

B.2 LIST_VOLUME or LIST_FILE OUTFILE record structures

The records are listed in COPY name alphabetical order.

B.2.1 DESC record (DE code)

This record is created by LIST_VOLUME to describe disk organization.

The “DE” record is always the last record written in OUTFILE.

H-DCT-FVL-DESC COPY contents :

```
*                       DISK ORGANIZATION
*
 01 CB-FVL-DESC.
    05 CB-CODE          PIC X(2).
*                                  /* CODE=DE                                  */
    05 CB-VERSION       PIC X(2).
*                                  /* VERSION                                  */
    05 CB-NBEFNMAX      PIC X(8).
*                                  /* NB MAX OF EFN IN VTOC (FBO ONLY)         */
    05 CB-NBEFNSEL      PIC X(8).
*                                  /* NB OF SELECTED FILES ACCORDING TO PREFIX  */
    05 CB-NBEFNARV      PIC X(8).
*                                  /* NB OF OMITTED FILES WITH ARVIOL          */
    05 CB-NBEFNEOR      PIC X(8).
*                                  /* NB OF OMITTED FILES WITH CATALOG ERROR    */
    05 CB-SZVTOC        PIC X(8).
*                                  /* SIZE OF VTOC (VBO) OR VTOCS (FBO)         */
    05 CB-SZVTOCP       PIC X(8).
*                                  /* SIZE OF VTOCP (FBO ONLY)                 */
    05 CB-SZUSED        PIC X(8).
*                                  /* TOTAL TRK/DBLK USED                      */
    05 CB-NBEXT         PIC X(8).
*                                  /* TOTAL OF EXTENTS FOR SELECTED FILES       */
    05 CB-SZFREE        PIC X(8).
*                                  /* TOTAL TRK/DBLK FREE                      */
    05 CB-NBFREE        PIC X(8).
*                                  /* NB OF FREE EXTENTS                       */
    05 CB-GTFREE        PIC X(8).
*                                  /* SIZE OF THE GREATEST FREE EXTENT          */
    05 CB-PCUSED        PIC X(3).
*                                  /* PERCENT USED ACCORDING TO PREFIX          */
    05 CB-QUOTAMAX      PIC X(8).
*                                  /* QUOTA MAX FOR THE PROJECT IN QUOTA UNIT   */
    05 CB-QUOTAUSED     PIC X(8).
*                                  /* QUOTA USED BY THE PROJECT IN QUOTA UNIT   */
    05 CB-QUOTAPC       PIC X(3).
*                                  /* PERCENT OF QUOTA USED ACCORDING TO PREFIX */
```

```
    05 CB-QUOTAUNIT     PIC X(8).
*                                  /* NB OF BYTES IN QUOTA UNIT                 */
    05 FILLER           PIC X(53).
```

B.2.2 ERROR record (ER code)

This record is created by LIST_VOLUME or LIST_FILE to give information about a possible error.

H-DCT-FVL-ERROR COPY contents :

```
*                        ERROR INFORMATION
*
 01 CB-FVL-ERROR.
    05 CB-CODE           PIC X(2).
*                                    /* CODE=ER             */
    05 CB-VERSION        PIC X(2).
*                                    /* VERSION             */
    05 CB-NUMERR         PIC X(4).
*                                    /* ERROR NUMBER        */
    05 CB-EDITRC.
       08 CB-RCEGAL      PIC X(3).
*                                    /* RC=                 */
       08 CB-RCHEX       PIC X(8).
*                                    /* RC IN HEXA          */
       08 CB-ARROW       PIC X(2).
*                                    /* ->                  */
       08 CB-SIGNAT      PIC X(8).
*                                    /* SIU                 */
       08 CB-COMMA       PIC X(1).
*                                    /* ,                   */
       08 CB-RCCHAR      PIC X(8).
*                                    /* MNEMONIC RC         */
    05 CB-WORDING        PIC X(44).
*                                    /* EXPLANATORY TEXT    */
    05 FILLER            PIC X(93).
```

B.2.3 ORG1 record (01 code)

This record is created by LIST_VOLUME or LIST_FILE to describe file general organization.

The “O1” record is written when the ORG option is required.

H-DCT-FVL-ORG1 COPY contents :

```
*                       FILE GENERAL ORGANIZATION
*
 01 CB-FVL-ORG1.
    05 CB-CODE          PIC X(2).
*                                        /* CODE=O1                 */
    05 CB-VERSION       PIC X(2).
*                                        /* VERSION                 */
    05 CB-FORMAT        PIC X(6).
*                                        /* FORMAT=                 */
    05 CB-ORGANIZATION  PIC X(8).
*                                        /* ORGANIZ=                */
    05 CB-CATALOG       PIC X(5).
*                                        /* "CAT  " "UNCAT" OR "ERR  "*/
    05 CB-RECFORM       PIC X(3).
*                                        /* RECFORM=                */
    05 CB-BLKSIZE       PIC X(6).
*                                        /* BLKSIZE=                */
    05 CB-CISIZE        PIC X(6).
*                                        /* CISIZE=                 */
    05 CB-RECSIZE       PIC X(6).
*                                        /* RECSIZE=                */
    05 CB-EXPDATE       PIC X(6).
*                                        /* EXPDATE=   ("YY/DDD")   */
    05 CB-RFU1          PIC X(4).
*                                        /* RFU1    =               */
    05 CB-UNIT          PIC X(3).
*                                        /* UNIT=                   */
    05 CB-DELREC        PIC X(3).
*                                        /* " DR" OR "NDR"          */
    05 CB-COMPACT       PIC X(3).
*                                        /* " CP" OR "NCP"          */
    05 CB-KEYSIZE       PIC X(4).
*                                        /* KEYSIZE=                */
    05 CB-KEYLOC        PIC X(6).
*                                        /* KEYLOC=                 */
    05 CB-FSN           PIC X(4).
*                                        /* FILE SERIAL NUMBER=     */
    05 CB-INCRSIZE      PIC X(6).
*                                        /* INCRSIZE=               */
    05 CB-BSN           PIC X(4).
*                                        /* " BSN" OR "NBSN"        */
    05 CB-CASIZE        PIC X(6).
```

```
*                                        /* CASIZE=                  */
    05 CB-CIFSP         PIC X(3).
*                                        /* CIFSP   =                */
    05 CB-CAFSP         PIC X(3).
*                                        /* CAFSP   =                */
    05 CB-COLLATE       PIC X(6).
*                                        /* COLLATE=                 */
    05 CB-DIRSIZE       PIC X(4).
*                                        /*DIRSIZE=                  */
    05 CB-LOGTRKSZ      PIC X(6).
*                                        /* LOGTRKSZ=                */
    05 CB-TYPE          PIC X(2).
*                                        /* TYPE=                    */
    05 CB-FIXTRACK      PIC X(4).
*                                        /* " FXT" OR "NFXT"         */
    05 CB-MASTER        PIC X(3).
*                                        /* " MT" OR "NMT"           */
    05 CB-CYLOV         PIC X(4).
*                                        /* CYLOV=                   */
    05 CB-COMPREC       PIC X(3).
*                                        /* " CR" OR "NCR"           */
    05 CB-IDXTYPE       PIC X(8).
*                                        /* IDXTYPE=                 */
    05 CB-RELOCAT       PIC X(3).
*                                        /* " RL" OR "NRL"           */
    05 CB-ONELOAD       PIC X(3).
*                                        /* " OL" OR "NOL"           */
    05 CB-LABEL-COMPACT PIC X(2).
*                                        /* "LC" OR "  "             */
    05 CB-MAXSIZE       PIC X(8).
*                                        /* MAXSIZE OF LIBRARY       */
    05 CB-UFMAXSIZE     PIC X(10).
*                                        /* MAXSIZE OF UFAS FILE     */
    05 FILLER           PIC X(10).
```

B.2.4 ORG2 record (02 code)

This record is created by LIST_VOLUME or LIST_FILE to describe secondary keys.

The “O2” record is written when the ORG option is required and the file is an UFAS one with secondary keys.

H-DCT-FVL-ORG2 COPY contents :

```
*                       SECONDARY KEY INFORMATION
*
 01 CB-FVL-ORG2.
    05 CB-CODE          PIC X(2).
*                                         /* CODE=O2                    */
    05 CB-VERSION       PIC X(2).
*                                         /* VERSION                    */
    05 CB-NOKEY         PIC X(2).
*                                         /* SEC.KEY NO=                */
    05 CB-KEYSIZE       PIC X(4).
*                                         /* SEC.KEY KEYSIZE=           */
    05 CB-KEYLOC        PIC X(6).
*                                         /* SEC.KEY KEYLOC=            */
    05 CB-DUPREC        PIC X(4).
*                                         /* " DUP" OR "NDUP"           */
    05 FILLER           PIC X(155).
```

B.2.5 SAVINFO record (SI code)

This record is created by LIST_FILE to give information about file or volume save.

The “SI” record is written when the SAVINFO option is required.

H-DCT-FVL-SAVINFO COPY contents :

```
*                       FILE OR VOLUME SAVE INFORMATION
*
 01 CB-FVL-SAVINFO.
    05 CB-CODE          PIC X(2).
*                                        /* CODE=SI  (SAVINFO)            */
    05 CB-VERSION       PIC X(2).
*                                        /* VERSION                       */
    05 CB-TYP-SAVE      PIC X(1).
*                                        /* "F"=FILSAVE  "V"=VOLSAVE      */
    05 CB-EFN           PIC X(44).
*                                        /* SAVED FILE NAME               */
    05 CB-MD1           PIC X(6).
*                                        /* IMAGE BEGINS ON <MD1>         */
    05 CB-MD2           PIC X(6).
*                                        /* IMAGE ENDS ON <MD2>           */
    05 CB-NAME          PIC X(44).
*                                        /* IMAGE NAME                    */
    05 CB-DATE          PIC X(14).
*                                        /* SAVE DATE                     */
    05 CB-DVC           PIC X(7).
*                                        /* FROM DVC                      */
    05 CB-VOLORG        PIC X(3).
*                                        /* FROM VOLORG                   */
    05 CB-ISN           PIC X(6).
*                                        /* IMAGE SERIAL NUMBER           */
    05 CB-SAV-DYN       PIC X(3).
*                                        /* "DYN"=DYNAMIC FILSAVE         */
    05 FILLER           PIC X(37).
```

B.2.6 SIZE record (SZ code)

This record is created by LIST_VOLUME or LIST_FILE to give file size information.

The “SZ” record is written when the SPACE option is required and the file is a multi-extent UFAS one. It gives the global size of the file.

H-DCT-FVL-SIZE COPY contents :

```
*                       FILE SIZE INFORMATION
*
 01 CB-FVL-SIZE.
    05 CB-CODE          PIC X(2).
*                                         /* CODE=SZ                */
    05 CB-VERSION       PIC X(2).
*                                         /* VERSION                */
    05 CB-TOTAL-SIZE    PIC X(8).
*                                         /* TOTAL SIZE=            */
    05 CB-TOTAL-UNIT    PIC X(3).
*                                         /* TOTAL UNIT=            */
    05 CB-TOTAL-SIZEPHY PIC X(8).
*                                         /* TOTAL SIZE IN DBLK ! TRK */
    05 FILLER           PIC X(152).
```

B.2.7 SPACE record (SP code)

This record is created by LIST_VOLUME or LIST_FILE to give space information.

The “SP” record is written when the SPACE option is required.

There is one record for each extent.

H-DCT-FVL-SPACE COPY contents :

```
*                       SPACE INFORMATION
*
 01 CB-FVL-SPACE.
    05 CB-CODE          PIC X(2).
*                                         /* CODE=SP                 */
    05 CB-VERSION       PIC X(2).
*                                         /* VERSION                 */
    05 CB-MD            PIC X(6).
*                                         /* MEDIA=                  */
    05 CB-EXT-SN        PIC X(3).
*                                         /* EXTENT-SN=              */
    05 CB-SIZE          PIC X(8).
*                                         /* SIZE=                   */
    05 CB-UNIT          PIC X(3).
*                                         /* UNIT=                   */
    05 CB-START         PIC X(8).
*                                         /* START=                  */
    05 CB-END           PIC X(8).
*                                         /* END=                    */
    05 CB-RFU1          PIC X(23).
*                                         /* RFU1                    */
    05 CB-FIRST-FREE    PIC X(8).
*                                         /* DK FIRST FREE SECTOR=   */
    05 CB-PC-USED       PIC X(3).
*                                         /* DK PERCENT USED=        */
    05 CB-SIZEPHY       PIC X(8).
*                                         /* SIZE IN DBLK ! TRK      */
    05 FILLER           PIC X(93).
```

B.2.8 TITRE record (TI code)

This record is created by LIST_VOLUME or LIST_FILE to describe the title information.

The “TI” record is always written and identifies the beginning of a set of records for one file. The end of a set is the next “TI” record or the end of OUTFILE.

On LIST_VOLUME an “ER” (Error) output record can appear before a “TI” record in the case of access right violation (SCATVIOL).

H-DCT-FVL-TITRE COPY contents :

```
*                       TITLE INFORMATION
*
 01 CB-FVL-TITRE.
    05 CB-CODE          PIC X(2).
*                                        /* CODE=TI                    */
    05 CB-VERSION       PIC X(2).
*                                        /* VERSION                    */
    05 CB-FILENAME      PIC X(44).
*                                        /* FILE :                     */
    05 CB-FILEOWNER     PIC X(12).
    05 CB-VOLORG        PIC X(3).
    05 FILLER           PIC X(112).
```

B.2.9 USA1 record (U1 code)

This record is created by LIST_VOLUME of LIST_FILE to describe general USAGE information.

The “U1” record is written when the USAGE option is required.

H-DCT-FVL-USA1 COPY contents :

```
*                       GENERAL USAGE INFORMATION
*
 01 CB-FVL-USA1.
    05 CB-CODE          PIC X(2).
*                                         /* CODE=U1                    */
    05 CB-VERSION       PIC X(2).
*                                         /* VERSION                    */
    05 CB-CREDATE       PIC X(6).
*                                         /* CREATION DATE=             */
    05 CB-GENSYS        PIC X(13).
*                                         /* GENERATING SYSTEM=         */
    05 CB-VSN1          PIC X(6).
*                                         /* VSN FIRST VOLUME=          */
    05 CB-VSEQN         PIC X(4).
```

```
*                                          /* VSEQN OF THIS VOL=       */
    05 CB-FORMATTED     PIC X(3).
*                                          /* " FT" OR "NFT"           */
    05 CB-LAST-CLOSE    PIC X(3).
*                                          /* " LC" OR "NLC"           */
    05 CB-TRACK-OVERFL  PIC X(3).
*                                          /* " TO" OR "NTO"           */
    05 CB-DIR-OVERFL    PIC X(3).
*                                          /* " DO" OR "NDO"           */
    05 CB-HARDWARE-KEYS PIC X(3).
*                                          /* " HK" OR "NHK"           */
    05 CB-IND-OVERFL    PIC X(3).
*                                          /* " IO" OR "NIO"           */
    05 CB-RFU1          PIC X(2).
*                                          /* NOT USED                 */
    05 CB-RECORD-TECHNIC PIC X(12).
*                                          /* RECORDING TECHNIC=       */
    05 CB-RECORD-DENSITY PIC X(4).
*                                          /* RECORDING DENSITY=       */
    05 CB-SECURITY      PIC X(12).
*                                          /* SECURITY=                */
    05 CB-FORMAT-SSF    PIC X(3).
*                                          /* "SSF" OR " "             */
    05 CB-JOB-WRITTING  PIC X(7).
*                                          /* JOB OR JOB-STEP WRITTING */
*                                          /* FILE=                    */
    05 CB-DK-SECTLG     PIC X(4).
*                                          /* SECTLG=                  */
    05 CB-INTERCHANGE   PIC X(8).
*                                          /* INTERCHANGE LEVEL=       */
    05 CB-FILE-ACCESS   PIC X(12).
*                                          /* FILE ACCESS=             */
    05 CB-TRANSFER      PIC X(3).
*                                          /* FILE MAY BE TRANSFERRED  */
*                                          /* (" TR" OR "NTR")         */
    05 CB-VERIFIED      PIC X(3).
*                                          /* " VF" OR "NVF"           */
    05 CB-COPIED        PIC X(3).
*                                          /* " CF" OR "NCF"           */
    05 CB-DK-VERSION    PIC X(1).
*                                          /* DK VERSION=              */
    05 CB-MT-VERSION    PIC X(2).
*                                          /* MT VERSION=              */
    05 CB-MT-GENER      PIC X(4).
*                                          /* MT GENER  =              */
    05 CB-USERBUILT     PIC X(12).
*                                          /* USER WHO BUILT           */
    05 FILLER           PIC X(32).
```

B.2.10 USAASI record (AS code)

This record is created by LIST_VOLUME or LIST_FILE to describe file address space information.

The “AS” record is written when the USAGE option is required and the file is an UFAS one.

H-DCT-FVL-USAASI COPY contents :

```
*                       FILE ADDRESS SPACE INFORMATION
*
 01 CB-FVL-USAASI.
    05 CB-CODE          PIC X(2).
*                                        /* CODE=AS                    */
    05 CB-VERSION       PIC X(2).
*                                        /* VERSION                    */
    05 CB-NUM           PIC X(2).
*                                        /* ADDRESS SPACE NUM=         */
    05 CB-NB-ALLOC      PIC X(8).
*                                        /* #ALLOCATED=                */
    05 CB-NB-FORMAT     PIC X(8).
*                                        /* #FORMATTED=                */
    05 CB-CI-TRK        PIC X(4).
*                                        /* #CI PER TRK=               */
    05 CB-CISIZE        PIC X(6).
*                                        /* CISIZE=                    */
    05 CB-NB-EXT        PIC X(3).
*                                        /* #EXT=                      */
    05 CB-DBSZ          PIC X(5).
    05 FILLER           PIC X(135).
```

B.2.11 USAKEY record (KY code)

This record is created by LIST_VOLUME or LIST_FILE to give information about secondary keys.

The “KY” record is written when the USAGE option is required and the file is an UFAS one with secondary keys.

H-DCT-FVL-USAKEY COPY contents :

```
*                        SECONDARY KEY INFORMATION
*
 01 CB-FVL-USAKEY.
    05 CB-CODE           PIC X(2).
*                                          /* CODE=KY                    */
    05 CB-VERSION        PIC X(2).
*                                          /* VERSION                    */
    05 CB-NUM            PIC X(2).
*                                          /* KEY NUM=                   */
    05 CB-IDX-LEVEL      PIC X(3).
*                                          /* #INDEX BY LEVEL=           */
    05 CB-IDX-ROOT       PIC X(3).
*                                          /* INDEX ROOT CI#=            */
    05 CB-IAU            PIC X(3).
*                                          /* INDEX ALLOC UNIT=          */
    05 CB-ENT-IDX        PIC X(3).
*                                          /* #ENTRIES BY INDEX=         */
    05 FILLER            PIC X(157).
```

B.2.12 USALIB record (LI code)

This record is created by LIST_VOLUME or LIST_FILE to give specific information about libraries.

The “LI” record is written when the USAGE option is required and the file is an library one.

H-DCT-FVL-USALIB COPY contents :

```
*                          LIBRARY SPECIFIC INFORMATION
*
 01 CB-FVL-USALIB.
    05 CB-CODE             PIC X(2).
*                                             /* CODE=LI                        */
    05 CB-VERSION          PIC X(2).
*                                             /* VERSION                        */
    05 CB-NB-LOG-TRK       PIC X(8).
*                                             /* #LOG.TRK=                      */
    05 CB-NB-USED          PIC X(8).
*                                             /* #USED=                         */
    05 CB-NB-FREE          PIC X(8).
*                                             /* #FREE=                         */
    05 CB-PERCENT-USED     PIC X(3).
*                                             /* PERCENT USED=                  */
    05 CB-FIRST-BAM-TRK    PIC X(3).
*                                             /* TRK FIRST BAM BLOCK (VBO)      */
    05 CB-FIRST-BAM-REC    PIC X(6).
*                                             /* REC FIRST BAM BLOCK (VBO)       /
    05 CB-FIRST-DIROV-TRK  PIC X(3).
*                                             /* TRK FIRST DIR IN OVERFL (VBO)  */
    05 CB-FIRST-DIROV-REC  PIC X(6).
*                                             /* REC FIRST DIR IN OVERFL (VBO)  */
    05 CB-NB-DIR-BLK       PIC X(6).
*                                             /* #DIR BLOCKS=                   */
    05 CB-NB-DIR           PIC X(6).
*                                             /* NB DIR. BAM. OVERFL BLK        */
    05 CB-SUBFILE-NB       PIC X(6).
*                                             /* CURRENT SUBFL NUMBER=          */
    05 CB-FIRST-BAM-BLK    PIC X(8).
*                                             /* FIRST BAM BLOCK (FBO)          */
    05 FILLER              PIC X(100).
```

B.2.13 USASTAT record (ST code)

This record is created by LIST_VOLUME or LIST_FILE to give statistical information.

The “ST” record is written when the USAGE option is required and the file is an UFAS one.

H-DCT-FVL-USASTAT COPY contents :

```
*                          STATISTICAL INFORMATION
*
 01 CB-FVL-USASTAT.
    05 CB-CODE             PIC X(2).
*                                           /* CODE=ST                     */
    05 CB-VERSION          PIC X(2).
*                                           /* VERSION                     */
    05 CB-S2-AS-NUM        PIC X(2).
*                                           /* ADDR SPACE NUM= "S2"        */
    05 CB-S2-ACTIVE        PIC X(8).
*                                           /* #ACTIVE CI(S2)=             */
    05 CB-S2-CI-SPLIT      PIC X(6).
*                                           /* #CI SPLITTING(S2)=          */
    05 CB-S2-CA-OR-IDX     PIC X(3).
*                                           /* "CA " OR "IDX"              */
    05 CB-S2-CA-IDX-SPLIT PIC X(6).
*                                           /* #CA SPLIT OR IDX REORG (S2) */
    05 CB-S2-ENT-IDX       PIC X(6).
*                                           /* #ENTRIES BY IDX(S2)=        */
    05 CB-S5-AS-NUM        PIC X(2).
*                                           /* ADDR SPACE NUM= "S5"        */
    05 CB-S5-ACTIVE        PIC X(8).
*                                           /* #ACTIVE CI(S5)=             */
    05 CB-S5-CI-SPLIT      PIC X(6).
*                                           /* #CI SPLITTING(S5)=          */
    05 CB-S5-CA-OR-IDX     PIC X(3).
*                                           /* "CA " OR "IDX"              */
    05 CB-S5-CA-IDX-SPLIT PIC X(6).
*                                           /* #CA SPLIT OR IDX REORG (S5) */
    05 CB-S5-ENT-IDX       PIC X(6).
*                                           /* #ENTRIES BY IDX(S5)=        */
    05 FILLER              PIC X(109).
```

B.2.14 USAUFAS record (UF code)

This record is created by LIST_VOLUME or LIST_FILE to give specific information about UFAS files.

The “UF” record is written when the USAGE option is required and the file is an UFAS one.

H-DCT-FVL-USAUFAS COPY contents :

```
*                       UFAS SPECIFIC INFORMATION
*
 01 CB-FVL-USAUFAS.
    05 CB-CODE          PIC X(2).
*                                              /* CODE=UF                    */
    05 CB-VERSION       PIC X(2).
*                                              /* VERSION                    */
    05 CB-STATUS        PIC X(8).
*                                              /* "STABLE" OR "UNSTABLE"     */
    05 CB-CI-FORMAT     PIC X(1).
*                                              /* DATA FORMAT CI=            */
    05 CB-FILE-VERSION  PIC X(2).
*                                              /* FILE VERSION NUM=          */
    05 CB-NB-USER-LAB   PIC X(4).
*                                              /* #USER LABELS=              */
    05 CB-MAX-TRK       PIC X(6).
*                                              /* FILE MAX TRK NUM=          */
    05 CB-MAX-CI        PIC X(8).
*                                              /* FILE MAX CI NUM=           */
    05 CB-USED-RATIO    PIC X(3).
*                                              /* USED RATIO                 */
    05 CB-NBREC         PIC X(10).
*                                              /* NB. OF RECORDS             */
    05 FILLER           PIC X(129).
```

B.2.15 USAUNS record (US code)

This record is created by LIST_VOLUME or LIST_FILE to give information about files in unstable state.

The “US” record is written when the USAGE option is required and the file is an UFAS one in unstable state.

For this COPY, NSTD level of Cobol is necessary.

H-DCT-FVL-USAUNS COPY contents :

```
*                        USAGE UNSTABILITY INFORMATION
*
 01 CB-FVL-USAUNS.
    05 CB-CODE          PIC X(2).
*                                         /* CODE=US                  */
    05 CB-VERSION       PIC X(2).
*                                         /* VERSION                  */
    05 CB-STATUS  OCCURS 8    PIC 1(8) BIT.
*                                         /* STATUS    = 1  OR  0     */
    05 CB-STATUS2 OCCURS 16   PIC 1(8) BIT.
*                                         /* STATUS2(I)= 1  OR  0     */
    05 FILLER           PIC X(147).
```

B.2.16 VOLUME record (V1 code)

This record is created by LIST_VOLUME to give volume information.

The “V1” record is always the first record written in OUTFILE.

H-DCT-FVL-VOLUME COPY contents :

```
*                       VOLUME INFORMATION
*
 01 CB-FVL-VOLUME.
    05 CB-CODE          PIC X(2).
*                                        /* CODE=V1                       */
    05 CB-VERSION       PIC X(2).
*                                        /* VERSION                       */
    05 CB-MD            PIC X(6).
*                                        /* MD=                           */
    05 CB-DVC           PIC X(44).
*                                        /* DVC=                          */
    05 CB-VOLORG        PIC X(3).
*                                        /* VBO ! FBO                     */
    05 CB-PROTECTED     PIC X(7).
*                                        /* PUBLIC ! PRIVATE ! QUOTA ! NONE */
    05 CB-OWNER         PIC X(12).
*                                        /* OWNER NAME                    */
    05 CB-PREFIX        PIC X(44).
*                                        /* PREFIX VALUE                  */
    05 CB-CAPACITY      PIC X(8).
*                                        /* VOLUME CONTENT IN TRK ! DBLK  */
    05 CB-DBLKSZ        PIC X(5).
*                                        /* DATA BLOCK SIZE IF FBO        */
    05 CB-PREPDATE.
*                                        /* PREPARATION DATE              */
       08 CB-YP         PIC X(2).
*                                        /* YEAR                          */
       08 CB-SLASH      PIC X(1).
*                                        /* SLASH CHARACTER               */
       08 CB-DDP        PIC X(3).
*                                        /* DAY IN YEAR                   */
    05 CB-MIRROR        PIC X(6).
*                                        /* MIRRORED DISK                 */
    05 CB-UNSTABLE      PIC X(8).
*                                        /* UNSTABLE DISK                 */
    05 CB-HRD           PIC X(3).
*                                        /* HRD VOLUME                    */
    05 CB-TYP-HRD       PIC X(3).
*                                        /* TYPE OF HRD VOLUME            */
    05 FILLER           PIC X(16).
```

❑

C. QUEUED files

This Appendix describes the COBOL procedure call, used to retrieve the user label of a queued subfile.

`H_QL_UGETLAB`	Get user label from a queued subfile

C.1 H_QL_UGETLAB

Programmatic Format

```
CALL "H_QL_UGETLAB" USING <queued_file_literal>,
QL-EXECUTION-RESULT,QL-USER-LABEL.
```

Structure of <queued_file_literal>

`queued_file_literal` is a data structure which identifies the "file literal" description of the subfile which contains the user label to be retrieved.

It must have the following format:

```
01 QUEUED-FILE-LITERAL.
  02 QUEUED-FILE-LITERAL-LENGTH COMP-1.
  02 QUEUED-FILE-LITERAL-STRING PIC X(QUEUED-FILE-LITERAL-LENGTH).
```

Where `QUEUED_FILE-LITERAL-LENGTH` is the size in bytes of the `QUEUED-FILE-LITERAL-STRING` item.

`queued_file_literal` being the local dps7_file_literal of the format:

`external_file_name..subfile_name [:media:devclass]`
where:
external_file_name: mandatory subfile_name: mandatory
media:devclass: not specified if the file is resident or cataloged.

Structure of QL-EXECUTION-RESULT

The following COBOL structure will receive values giving the Results of the Execution.

```
01 QL-EXECUTION-RESULT.
   02 QL-RESULT COMP-2.
   02 QL-RC COMP-2.
```

QL-RESULT *output parameter:* a number to identify the result of the execution of the programmatic interface

0 - successful completion of the procedure
1 - unsuccessful completion of the procedure.
     Reasons may be :
          - erroneous input parameter
          - output parameters not writable
2 - unsuccessful completion of the procedure.
          - File Definition Creation failed
3 - unsuccessful completion of the procedure.
          - File Assignement failed
4 - unsuccessful completion of the procedure.
          - File Open failed
5 - unsuccessful completion of the procedure.
          - Subfile Opens failed

QL-RC *output parameter:* a number to identify the return code of the execution of the programmatic interface

QL-RC can be edited in the following way:

```
01 EDITRC PIC X (30).
CALL "H_STD_UEDTG4" USING EDITRC ADDRESS OF QL-RC.
DISPLAY "RETURN CODE=" EDITRC.
```

QL-RC is only significant for values 2,3,4,5 of **QL-RESULT**

Structure of QL-USER-LABEL

The following COBOL structure will receive the user label.

```
01 QL-USER-LABEL.
   02 QL-LAB-LENGTH PIC 1(8) USAGE BIT VALUE B"11111111".
   02 QL-LAB-DATA PIC X(255).
```

`QL-LAB-LENGTH`	When used for reading a user label, it initially indicates the space available for the label data, and is set to the size of the label data actually presented to the user. The `QL-LAB-LENGTH` is set to zero if there is no label data for the subfile.
`QL-LAB-DATA`	The label data associated with the subfile. Its length should normally be declared as 255 characters which is the maximum allowed.

C.2 Copy file for user label contents of subfile created by WRITER QUEUED

The name of this COPY file is **OW-QL-USER-LABEL** and is stored in the **SYS.HSLLIB**

```
*   USER LABEL OF WRITER QUEUED SUBFILE
*
 01 OW-QL-USER-LABEL.
    02 OW-QL-USER-LABEL-LENGTH   PIC 1(8) USAGE BIT   VALUE B"11111111".
    02 OW-QL-USER-LABEL-INFO.
       03 OW-QL-RUN-INFO.
          04 OW-QL-MBZ           COMP-2.
*                                    /* MBZ                       */
          04 OW-QL-RON           PIC X(4).
*                                    /* RON OF THE JOB            */
          04 OW-QL-JOBID         PIC X(8).
*                                    /* JOB IDENTIFICATION        */
          04 OW-QL-BILLING       PIC X(12).
*                                    /* ACCOUNTING IDENTIFICATION */
          04 OW-QL-PROJECT       PIC X(12).
*                                    /* PROJECT IDENTIFICATION    */
          04 OW-QL-USER          PIC X(12).
*                                    /* USER IDENTIFICATION       */
          04 OW-QL-INT-DATE      PIC X(5).
*                                    /* INTRODUCTION DATE         */
          04 OW-QL-INT-TIME      PIC X(6).
*                                    /* INTRODUCTION TIME         */
* --------- INFORMATION ON THE JOB SUBMITTER -----------
       03 OW-QL-SUBMITTER-INFO.
          04 OW-QL-SUB-HOST      PIC X(8).
*                                    /* HOST IDENTIFICATION       */
          04 OW-QL-SUB-STATION   PIC X(8).
*                                    /* STATION IDENFICATION      */
          04 OW-QL-SUB-OPERATOR  COMP-2.
*                                    /* OPERATOR IDENTIFICATION   */
          04 OW-QL-SUB-USER      PIC X(12).
*                                    /* SUBMITTER IDENTIFICATION  */
          04 OW-QL-SUB-BILLING   PIC X(12).
*                                    /* BILLING IDENTIFICATION    */
          04 OW-QL-SUB-PROJECT   PIC X(12).
*                                    /* PROJECT IDENTIFICATION    */
* -------------- DESCRIPTION OF THE SYSOUT -------------
       03 OW-QL-SYSOUT-INFO.
          04 OW-QL-NAME          PIC X(8).
*                                    /* OUTPUT NAME               */
          04 OW-QL-CLASS         PIC X(1).
*                                    /* OUTPUT CLASS              */
          04 OW-QL-PRIORITY      PIC 1(8) USAGE BIT.
*                                    /* OUTPUT PRIORITY           */
          04 OW-QL-DEST-HOST     PIC X(8).
*                                    /* OUTPUT DESTINATION        */
          04 OW-QL-DEST-STATION  PIC X(8).
*                                    /* OUTPUT DESTINATION        */
          04 OW-QL-COPY-NB       PIC 1(8) USAGE BIT.
*                                    /* NUMBER OF COPIES          */
          04 OW-QL-BANNER-INFO.
             05 OW-QL-BANNER-TXT1 PIC X(12).
*                                    /* BANNER INFORMATION        */
             05 OW-QL-BANNER-TXT2 PIC X(12).
*                                    /* BANNER INFORMATION        */
             05 OW-QL-BANNER-TXT3 PIC X(12).
*                                    /* BANNER INFORMATION        */
             05 OW-QL-BANNER-TXT4 PIC X(12).
*                                    /* BANNER INFORMATION        */
          04 OW-QL-MEDIA.
*                                    /* MEDIA IDENTIFICATION      */
             05 OW-QL-BELT        PIC X(2).
*                                    /* BELT IDENTIFICATION       */
             05 OW-QL-PAPER       PIC X(4).
*                                    /* PAPER IDENTIFICATION      */
          04 OW-QL-SEQ-INDEX     COMP-1.
*                                    /* OUTPUT NUMBER             */
          04 OW-QL-JOBOUT-INDEX  COMP-1.
*                                    /* OUTPUT NUMBER IF JOBOUT   */
          04 OW-QL-REC-NUMBER    COMP-2.
*                                    /* SIZE OF THE OUTPUT        */
          04 OW-QL-PAGE-NB       COMP-2.
*                                    /* NUMBER OF PAGES OF OUTPUT */
          04 OW-QL-DEV-TYPE      PIC X(2).
*                                    /* DEVICE TYPE               */
          04 OW-QL-DEV-ATTR      COMP-1.
*                                    /* DEVICE ATTRIBUTES         */
    02 OW-QL-DISP              PIC 1(8) USAGE BIT.
*                                    /* DISP FLAG                 */
    02 OW-QL-SSF-RECORDS-NB    COMP-2.
*                                    /* NB OF SSF RECORDS          */
    02 OW-QL-RFU1              PIC X(33).
    02 OW-QL-OWNER             PIC 1(8) USAGE BIT.
```

❑

Index

Technical publication remarks form

Title :	DPS7000/XTA NOVASCALE 7000 System Call from COBOL Languages: COBOL

Reference N° :	47 A2 04UL 05	**Date:**	May 2002

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be promptly investigated by qualified technical personnel and action will be taken as required.
If you require a written reply, please include your complete mailing address below.

NAME : ____________________ Date : __________

COMPANY : ____________________

ADDRESS : ____________________

____

Please give this technical publication remarks form to your BULL representative or mail to:

Bull - Documentation D$^{ept.}$
1 Rue de Provence
BP 208
38432 ECHIROLLES CEDEX
FRANCE
info@frec.bull.fr

Technical publications ordering form

To order additional publications, please fill in a copy of this form and send it via mail to:

BULL CEDOC **357 AVENUE PATTON** **B.P.20845** **49008 ANGERS CEDEX 01** **FRANCE**	**Phone:** **FAX:** **E-Mail:**	+33 (0) 2 41 73 72 66 +33 (0) 2 41 73 70 66 srv.Duplicopy@bull.net

CEDOC Reference #	Designation	Qty
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
_ _ _ _ _ _ _ _ _ [_ _]		
[_ _] : The latest revision will be provided if no revision number is given.		

NAME: ______________________________ Date: __________

COMPANY: ______________________________

ADDRESS: ______________________________

____

PHONE: ____________________ FAX: ____________________

E-MAIL: ______________________________

For Bull Subsidiaries:

Identification: ______________________________

For Bull Affiliated Customers:

Customer Code: ______________________________

For Bull Internal Customers:

Budgetary Section: ______________________________

For Others: Please ask your Bull representative.

BULL CEDOC
357 AVENUE PATTON
B.P.20845
49008 ANGERS CEDEX 01
FRANCE

REFERENCE
47 A2 04UL 05