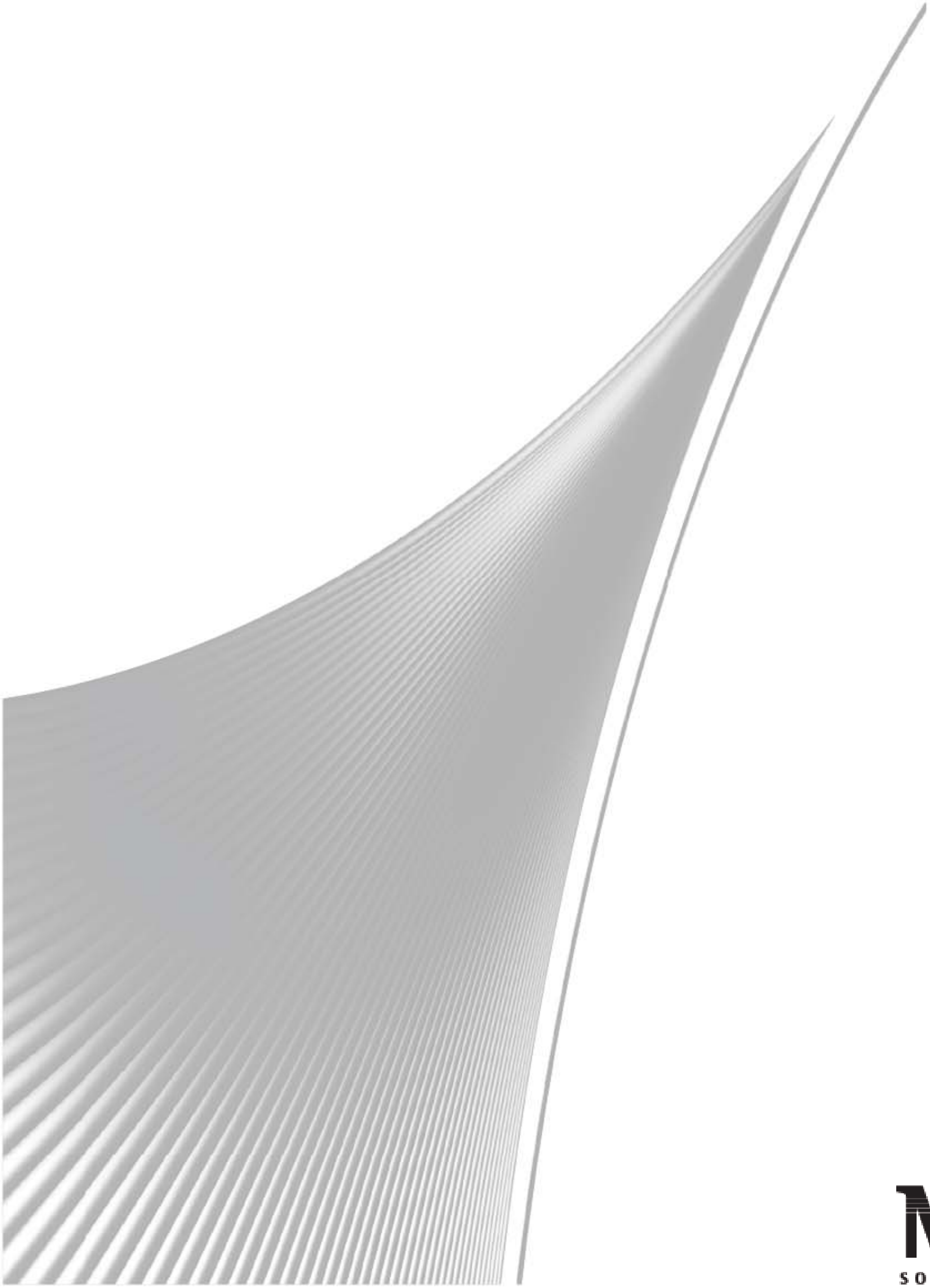
ThruPut Manager®

Base Product

System Programming Guide



MVS
solutions inc.

This manual applies to ThruPut Manager®Version 7 Release 1.0 PTF level 7109 and to all further releases until otherwise indicated. This material contains proprietary information on ThruPut Manager. Use of this manual is subject to the terms of the user’s software license agreement for ThruPut Manager.

The complete list of manuals relevant to this release is available from the Customer Center on www.mvssol.com, from where any of the manuals may be downloaded.

The Acrobat® PDF Format

This set of manuals has been formatted to make it suitable for soft copy as well as hard copy formats. If reading the manual online, note that:

- Entries in the Table of Contents are hot-linked and can be used for document navigation.
- Other cross-references appear in blue type and are also hot-linked for easy navigation.

For best results, turn on the Bookmarks feature in your Acrobat® PDF reader.

Preface

About This Manual

This manual provides information for systems programmers involved in planning, installing, and customizing ThruPut Manager. There are five system programming manuals as listed below. Information described in other manuals is referred to by guide name and chapter as a quick and easy cross reference.

- Dataset Contention Services System Programming Guide
- Drive Booking Services (DBS) System Programming Guide
- Job Binding Services (JBS) System Programming Guide
- User Control Services (UCS) System Programming Guide
- Exits System Programming Guide

Summary of Changes

V7R1-7109

(April 2017)

- No changes.

V7R1-7108

(February 2017)

- No changes.

V7R1-7107

(May 2016)

- No changes.

V7R1-7106

(November 2015)

- Added SAF security system capability to restrict authority at the function or command level - Authorizing the Use of Operating Commands.

V7R1-7104

(July 2015)

- No changes.

V7R1-7101

(July 2014)

- This is a base manual for ThruPut Manager Version 7 Release 1.1.

Table of Chapters

Preface

ThruPut Manager Base Product

General Services

Language Services

The Job Classing Component

The Job Setup Services Component

The SYSOUT Services Component

Index

Detailed Table of Contents

Preface

ThruPut Manager Base Product

Chapter 1. ThruPut Manager Base Product Functions—Your Playground

General Services

Chapter 2. TMSS Initialization Statements

Chapter 4. Printing Services

Language Services

Chapter 15. Table (TBL) Function

The Job Classing Component

Chapter 16. Job Selection Function

Chapter 17. Job Display Services (JOB) Function

Chapter 18. Deferred Processing Services (DPS) Function

Chapter 19. Job Validation Services (JVS)

Chapter 20. CPU Capping and Normalizing

Chapter 21. Volume Status (VOL) Function

Chapter 22. ThruPut Manager (TM) Function

The Job Setup Services Component

Chapter 23. JSS Hold Extensions

Chapter 24. JAL Extensions for JSS

Chapter 25. TM/JSS Support in the TMISPF Application

Chapter 26. StorageTek Automated Cartridge System (ACS) Function

Chapter 27. Memorex Telex 5400 Automated Tape Library (ATL) Function

Chapter 28. DFSMS Tape Storage Groups (SMS) Function

Chapter 29. Virtual Tape System (VTS) Support

Chapter 30. Dataset Recall Services (DRS)

The SYSOUT Services Component

Chapter 31. SYSOUT Services (SOS) Function

Index

ThruPut Manager Base Product

Chapter 1. ThruPut Manager Base Product Functions—Your Playground

This chapter describes the set of functions that the ThruPut Manager Base Product provides your installation to system manage batch. They are grouped under their corresponding components.

ThruPut Manager Base Product

Two components are included with the ThruPut Manager Base Product:

1. Job Classing
2. Job Setup

The Base Product also includes a number of General Services and Language Services that provide the infrastructure needed by all other ThruPut Manager components.

The following sections discuss the two components, the functions included in each of them, and the language and general services included with the ThruPut Manager Base Product.

Job Classing Component

The Job Classing component of ThruPut Manager provides all the functions required to automate the classification of batch work. In addition to the facilities needed for Job Classing, the Base Product provides the infrastructure needed by all other components.

The following functions are provided:

- Job Selection Function.
- Job Display Services (JOB) Function.
- User Display Facility (UDF).
- Deferred Processing Services (DPS) Function.
- Job Validation Services (JVS) Function.
- CPU Capping and Normalizing (CAP) Function.
- Volume Status (VOL) Function.

- ThruPut Manager (TM) Function.

Each function is described briefly below.

Job Selection Function

ThruPut Manager provides you with facilities to:

- Select the classes that are to receive ThruPut Manager processing. Alternatively, you can define the classes that are to be excluded.
- Define the classes used to provide Deferred Processing Services.

A description of this function is provided in ***Installation Guide*** " In this guide, see also "Chapter 16. Job Selection Function."

Job Display Services (JOB) Function

ThruPut Manager provides you with the ability to display information about a particular job. This information is collected by the Job Analyzer and made available through JAL for display purposes. Your installation can determine exactly what information is available for display, and the format the display will take. This control is provided at the job level, giving maximum flexibility.

The information is retained in the JES2 spool (together with the job) and contains the data which were used to determine the job's execution class, as well as a list of the volumes required by the job.

Operators can use Job Display Services (JOB) commands to help them in situations that require intervention.

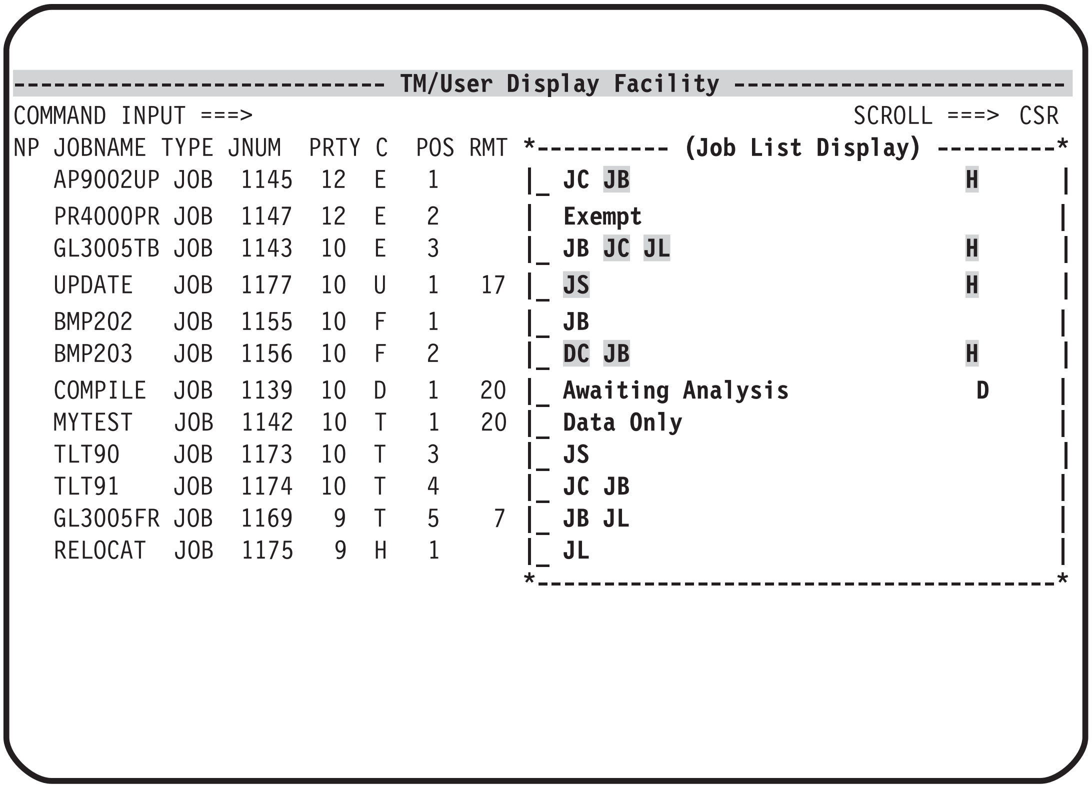
This function is described in "Chapter 17. Job Display Services (JOB) Function."

User Display Facility (UDF)

Information about jobs that are managed by ThruPut Manager can also be made available for display by end users through a windowed ISPF interface to ThruPut Manager.

When you are using a supported spool manager product, the TMUSER command presents a summary of jobs in a list window. This summary provides general information, including which ThruPut Manager applications are managing which jobs. Users can then select individual jobs and request specific information for each ThruPut Manager application that is managing the job.

The example on the following page shows a typical summary list of jobs, from which a user can select individual jobs for more detailed displays.



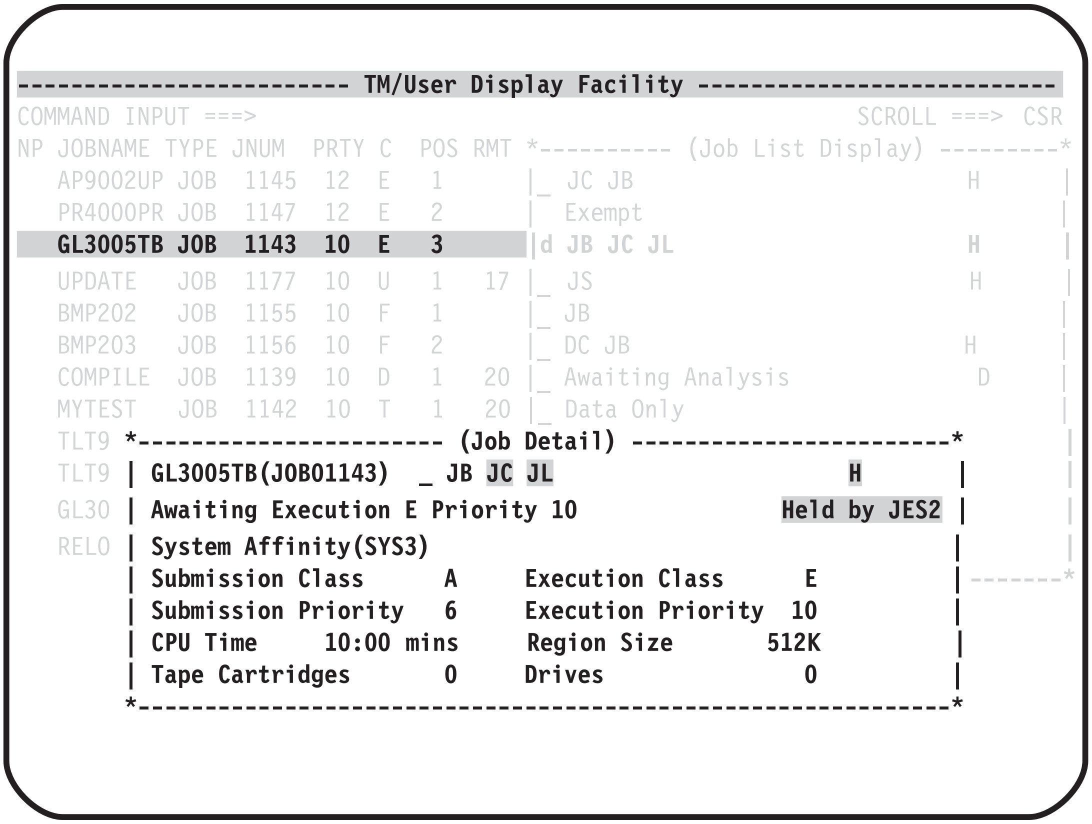
```
---------------------------- TM/User Display Facility -------------------------
COMMAND INPUT ===>                                                SCROLL ===>  CSR
NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                        H       |
   PR4000PR JOB  1147  12  E   2      |  Exempt                               |
   GL3005TB JOB  1143  10  E   3      |_ JB JC JL                     H       |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                           H       |
   BMP202   JOB  1155  10  F   1      |_ JB                                   |
   BMP203   JOB  1156  10  F   2      |_ DC JB                        H       |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis             D      |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                            |
   TLT90    JOB  1173  10  T   3      |_ JS                                   |
   TLT91    JOB  1174  10  T   4      |_ JC JB                                |
   GL3005FR JOB  1169   9  T   5    7 |_ JB JL                                |
   RELOCAT  JOB  1175   9  H   1      |_ JL                                   |
                                      *---------------------------------------*
```

Job List Window

You also have the option of providing a display of installation-specific job information. Through JAL, you can determine what details to make available to the user.

You can control the display's format, and you can limit job detail displays to users with appropriate authority. See the following page for a sample Job Detail display window.

The User Display Facility is described in "Chapter 5. User Display Facility (UDF)."



```
------------------------- TM/User Display Facility ---------------------------
COMMAND INPUT ===>                                              SCROLL ===>  CSR
NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                      H         |
   PR4000PR JOB  1147  12  E   2      |  Exempt                               |
   GL3005TB JOB  1143  10  E   3      |d JB JC JL                   H         |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                         H         |
   BMP202   JOB  1155  10  F   1      |_ JB                                   |
   BMP203   JOB  1156  10  F   2      |_ DC JB                      H         |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis           D        |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                            |
   TLT9 *----------------------- (Job Detail) ------------------------*       |
   TLT9 | GL3005TB(JOB01143)  _ JB JC JL                     H        |       |
   GL30 | Awaiting Execution E Priority 10              Held by JES2 |       |
   RELO | System Affinity(SYS3)                                      |       |
        | Submission Class      A     Execution Class      E        |-------*
        | Submission Priority   6     Execution Priority  10        |
        | CPU Time     10:00 mins     Region Size       512K        |
        | Tape Cartridges       0     Drives               0        |
        *------------------------------------------------------------*
```

Job Detail Window

Deferred Processing Service (DPS) Function

ThruPut Manager helps your installation to offer deferred batch services, such as overnight or weekend. For this type of service, it is best to check for JCL errors immediately. In contrast, the actual classing of the job should be done as close to run time as possible, using the latest catalog information.

To deal with this, ThruPut Manager lets you define a Deferred Job Analysis Class. Jobs eligible for Deferred Processing are first checked for JCL errors (Early Scan), and if no errors are found, are placed in the Deferred Analysis Class. A second ThruPut Manager pass, closer to the time the jobs are selected for execution, is possible by activating the appropriate deferred class.

This function is described in “Chapter 18. Deferred Processing Services (DPS) Function.”

Job Validation Services (JVS) Function

Jobs submitted for execution receive JCL checking during Job Analysis, detecting JCL errors that would otherwise cause the job to fail at execution time. You can extend this JCL checking to include the following types of jobs:

- Jobs submitted with TYPRUN=SCAN.
- Jobs submitted with TYPRUN=HOLD.

In a standard system, the first type of jobs receive a very cursory JCL checking, where gross errors can go undetected. Your installation can chose to give these jobs full JCL checking.

Jobs that are submitted with TYPRUN=HOLD do not receive full JCL checking until execution time. With ThruPut Manager, your installation can chose to give these jobs full JCL checking at submission time.

Note that jobs that are submitted for execution (and are not exempted) receive comprehensive JCL checking before they are queued for execution by ThruPut Manager.

This function is described in “Chapter 19. Job Validation Services (JVS).”

CPU Capping and Normalizing (CAP) Function

ThruPut Manager has facilities for your installation to provide a Job CPU limit for a particular job or group of jobs. This is done with the JAL action statement SET with the CPUCAP parameter. In addition, the CPU values can be “normalized” to adjust for different CPU speeds in a MAS complex.

This function is described in “Chapter 20. CPU Capping & Normalizing.”

Volume Status (VOL) Function

This function allows you to instruct ThruPut Manager to treat DASD volumes in a particular way, regardless of the actual status of the volumes in the system. It provides facilities to manage a DASD Volume List so you can associate status such as “not available”, or “resident” to a particular volume serial number.

Normally this function is used to handle exception conditions, for example when resident volumes become temporarily unavailable. You can instruct ThruPut Manager to hold jobs that reference the unavailable online DASD volumes.

This function is described in “Chapter 21. Volume Status (VOL) Function.”

ThruPut Manager (TM) Function

This function groups commands and services that serve a general purpose. Normally they operate on “environment” related items, such as the activation and deactivation of exits.

The following control mechanisms are provided under the TM function heading:

- Activating/deactivating ThruPut Manager User Exits.
- Activating/deactivating ThruPut Manager tracing.
- Defining how ThruPut Manager should treat certain unit names.
- Defining size of user data areas for exits.
- Batching ThruPut Manager commands in a file to be executed at initialization time.
- Displaying and altering ThruPut Manager Selection and Exempt classes.

This function is described in “Chapter 22. ThruPut Manager (TM) Function.”

Job Setup Services Component

The Job Setup Services (JSS) component of ThruPut Manager provides a group of comprehensive system managed functions to simplify the handling of jobs with tape requirements. JSS is the overall name for the following functionality:

- JSS Hold Extensions.
- Job Action Language (JAL) Extensions for JSS.
- User Display Facility (UDF) JSS Extensions.
- Automated Library Services:
  - StorageTek ACS Support Services.
  - Memorex Telex 5400 ATL Support Services.
  - IBM 3495 ATLDS Support Services.
- Dataset Recall Services (DRS).

The following sections give a brief description of the functions provided by this component. Each section points to the chapter that describes the function in detail.

JSS Hold Extensions

The Job Setup Services component makes use of the Control File Manager and Control File architecture to support new job-hold categories. The JSS application provides the necessary facilities to automatically hold/release jobs with setup requirements. In addition to the standard MVS/JES2 hold category, JSS allows jobs to be held in three additional categories:

- VOLUME hold, for jobs awaiting setup volumes.
- HSM hold, for jobs that have been delayed to allow time for DFHSM processing to complete.
- OPERATOR hold, a category that is controlled by operator commands.

For a description of the automated hold/release capabilities of JSS refer to “Chapter 23. JSS Hold Extensions.”

Job Action Language (JAL) Extensions for JSS

Extensions to JAL are provided to support Job Setup Services. The following actions are possible with the JSS extensions:

- Control the VOLUME Hold mechanism with Minimum and Maximum Hold Time requests.
- Initiate early dataset recalls for DFHSM migrated datasets.
- Specify a time delay for jobs that require dataset recalls.
- Extensions to the resource analysis process to support automated tape cartridge library systems such as the StorageTek ACS System.

A brief description of the JSS JAL extensions is provided in “Chapter 24. JAL Extensions for JSS.”

For a complete description, see the manual ***DAL/JAL User Guide***.

Automated Library Services

Automated Library Services allow your installation to distinguish between cartridges that have to be manually mounted and cartridges that are under the control of an automated library system. To determine the status of a given volume, ThruPut Manager’s Automated Library support makes a call to the API provided by the software supporting your automated library system.

ThruPut Manager provides support for:

- StorageTek Automated Cartridge System. This is described in "Chapter 26. Storagetek Automated Cartridge System (ACS) Function."
- Memorex Telex 5400 Automated Tape Library. This function is described in "Chapter 27. Memorex Telex 5400 Automated Tape Library (ATL) Function."
- IBM 3495 Automated Tape Library Dataserver. This function is described in "Chapter 28. DFSMS Tape Storage Groups (SMS) Function."

Dataset Recall Services (DRS)

Installations that have DFHSM can use DRS to recall datasets at Job Analysis time. The recall is asynchronous, that is, the recall is issued and you can specify how long the job is to be kept in HSM Hold, but the job is not released automatically when the dataset has been recalled, nor is the job kept in hold beyond the specified time if the dataset is not yet recalled.

The HSM recall facility can also be used with ABR at the V5.2 Level 30 or above.

This function is described in "Chapter 30. Dataset Recall Services (DRS)."

General Services

In order to support the components of ThruPut Manager, a number of General Services are included in the base product. These are:

- TMSS Initialization Services.
- File Definition Services.
- Printing Services
  - Console Printing Services.
  - SYSOUT Printing Services.
- JECL Services.
- Job Extract Facility (JEF).
- Job Analysis Affinity (JAF) Facility.
- Installation Exits Services.

Each of these services is explained briefly below.

☞ ***The General Services are described ahead of any other functions in the detailed section of this manual.***

TMSS Initialization Services

TMSS initialization services allow you to use initialization statements to configure ThruPut Manager to suit your installation. Using initialization statements, you can:

- Indicate which set of JAL rules ThruPut Manager will use, and where it is to be found.
- Specify which files are to be used, and their characteristics. (See the heading "File Definition Services", below.)
- Activate JAL tracing for JAL and/or ThruPut Manager processing.
- Submit a batch of commands that will be issued at TMSS initialization time.
- Select which type of Printing Service to use, and define its characteristics.
- Define installation exits.
- Allocate an area for each job and/or volume, for use in your installation exits.
- Define the unit names or storage groups that are to be handled by automated library support.
- Specify unit name changes that might occur after ThruPut Manager processing, allowing ThruPut Manager to take this into account when determining a Job Profile.
- Set the status of DASD volumes for the purposes of Job Analysis, overriding the volumes' actual status.

Additionally, simple control statements allow you to share the initialization statements across systems with different ThruPut Manager configurations.

The statements and their syntax are described in full in "Chapter 2. TMSS Initialization Statements."

File Definition Services

ThruPut Manager requires the Control File and the Volume Information File, and could also require the SPOOL File and the CMF File, depending on the options used at your installation.

File Definition Services includes the following:

- TMSS initialization statements, as described above, are used to define the files to be used.
- TMSS start-up parameters, entered on the START command, control formatting (cold-starting) the files.

- A utility is provided to move the Control File if necessary.

File Definition Services and file characteristics are described in “Chapter 3. File Definition Services (FDS) Function.”

Printing Services

ThruPut Manager provides two services to print output such as Volume Lists and DCS Alerts:

- **Console Printing Services (CPS)** allows you to use hard-copy console devices for printed output. Writers are provided so that output can be printed. Options are provided in the definition of CPS Writers to accommodate your installation needs.

  Output can be “spooled” using the SPOOL File. This file is designed to be shared by ThruPut Managers running on different CPUs. As a result you can share one CPS printer among all MAS systems.

- **SYSOUT Printing Services (SPS)** allows you to use standard JES2 SYSOUT for printed output. Standard SYSOUT characteristics, such as class and destination, can be defined and grouped together using a SYSOUT definition, which can then be selected in JAL for printing purposes.

For a complete description of Printing Services, refer to “Chapter 4. Printing Services.”

JECL Services

ThruPut Manager includes support that allows you to specify certain control statements through JECL. The same support for started tasks is provided through the DD subsystem interface. Additionally, where it is reasonable to do so, control statements are supported inside JCL procedures.

For a complete description of JECL Services, refer to "***JECL Reference Guide***".

Job Extract Facility (JEF)

Occasionally, your installation might want to obtain information that ThruPut Manager has collected for a particular job during the Job Analysis phase. The Job Extract Facility provides services so you can request that certain information be returned to you (the caller) about the job currently executing in your address space. You can make a request either from a program or from an SMF exit.

JEF has been designed to allow you to include ThruPut Manager’s information in your SMF records at job execution time.

This function is described in “Chapter 9. Job Extract (JEF) Facility.”

Job Analysis Affinity Function (JAF)

The Job Analysis Affinity facility of ThruPut Manager allows an installation to specify a system, or systems, affinity default in a MAS complex for the ***conversion and analysis*** process. The default is applicable to the jobs entering a system where this facility (AFFINITY) has been requested.

In a standard JES2 system, the default value is ANY and no facilities are provided to change it.

This function is described in "Chapter 8. Job Analysis Affinity."

Installation Exit Services

ThruPut Manager provides a comprehensive set of installation exit points to allow you to accommodate any special circumstances that prevail in your installation. These exits let you implement interfaces systems such as your accounting, security, and library systems.

A set of mapping macros for the installation exit parameter lists is included with ThruPut Manager.

For a description of Installation Exit Services, see "Chapter 11. Installation Exits Summary."

For a complete description of installation exits, refer to the publication ***Exits: System Programming Guide***.

Language Services

In order to support the components of ThruPut Manager, a number of Language Services are included in the base product. These are:

- TM Detail Action Language (DAL) Function.
- Job Action Language (JAL) Function.
- Table Support Facility (TBL).

TM Detail Action Language (DAL) Function

ThruPut Manager executes your TM DAL statements for every DD statement encountered in a job. To maintain integrity and for your convenience, a separate Language Processor is provided so that your DAL is verified and converted to ThruPut Manager's internal format before installing a new set of rules.

Using ***TM DAL***, you can verify DD statement parameters, issue messages, and if necessary, fail jobs. You can also set User Logic Variables to communicate with JAL and subsequent invocations of DAL.

A description of this function is provided in "Chapter 14. Detail Action Language (DAL) Function."

For a complete description, see the manuals ***DAL/JAL User Guide*** and ***JAL Reference Guide***.

Job Action Language (JAL) Function

JAL is the reason ThruPut Manager can be such a powerful tool. JAL allows your installation to define rules for the automation of job classing, directing ThruPut Manager to take actions such as:

- Classing jobs.
- Establishing job priorities.
- Cancelling a job.
- Setting a job's JES2 HOLD status.
- CPU Capping and Normalizing.
- Inserting messages into JOBLOG/SYSMSGS.

ThruPut Manager executes your JAL statements each time a job is selected. To maintain integrity and for your convenience, a separate Language Processor is provided so that your JAL is verified and converted to ThruPut Manager's internal format before installing a new set of rules. JAL provides the power to determine what actions to take for each job based on the rules your installation chooses.

A description of this function is provided in "Chapter 13. Job Action Language (JAL) Function."

For a complete description, see the manuals ***DAL/JAL User Guide*** and ***JAL Reference Guide***.

Table Support Facility (TBL)

The Table Support Facility provides an ***external*** way to manage groups of values that are used to verify Job Descriptor values in JAL or DAL. Table Support allows you to place these values in tables, and also makes the management of such tables external. You can refresh tables "on-the-fly" and avoid the need to change JAL or DAL just to add or change a value.

For example, your installation might have a large group of account codes that are not suitable for pattern matching. Without the Table Support Facility, changing a single value would mean re-processing your DAL or JAL. With Table Support, this is reduced to adding an entry in a table, which can be then be refreshed.

The TBL command function provides operator commands to refresh, activate, and deactivate JAL/DAL tables, as well as display their status.

This function is described in "Chapter 15. Table (TBL) Function."

SYSOUT Services

With SYSOUT Services (SOS), you can examine SYSOUT DD statements and change them if desired. Using DAL, you can check for compliance with standards and alter the SYSOUT statement if necessary. You can also accommodate installation changes such as new printing devices or print classes without requiring that users change JCL.

SOS can alter anything that a user can request on a SYSOUT DD statement or a JES2 /*OUTPUT statement. For a discussion of this facility, refer to "Chapter 31. SYSOUT Services (SOS) Function."

General Services

Chapter 2. TMSS Initialization Statements

This chapter describes the FOR Group, the syntax for TMSS initialization statements, and each statement in detail.

TMSS Initialization Statements

There are a number of TMSS initialization statements that allow you to customize ThruPut Manager to your environment.

These statements provide TMSS with the information needed to create a framework to interpret the rules that you define with the Job Action Language.

An example illustrates the point: the ‘JAL LOAD dataset-name’ initialization statement tells ThruPut Manager where to find the Job Action Language that you have written. At initialization time, TMSS brings the JAL (your rules) from the specified dataset.

Associating TMSS Statements with a System

To facilitate sharing common TMSS initialization statements for multiple systems, the “FOR Group” facility is provided. Use this facility only when you have unique statements for a particular system. The format is as follows:

```
FOR SMFID(sys[,sys,...])
      statement
      statement
      ...
ENDFOR
```

Where ***sys*** represents the SMFID of a system that should honor the TMSS initialization statements that follow.

To terminate the FOR condition, use the ENDFOR statement.

The rules for using the FOR group facility are as follows:

- Statements that are not within a “FOR Group” are always applicable regardless of the system that processes them.
- Nested FOR groups are not allowed. If another FOR statement is encountered before an ENDFOR terminates the previous group, it is treated as an error.

- You can specify a list of SMFIDs, separated by commas.
- The FOR and ENDFOR statements can begin in any column of the record. Only columns 1 through 72 are scanned.
- If the SMFID of the system processing a FOR Group does not match the FOR SMFID, the statements are ignored—almost. The statements receive simple syntax checking, so they must conform to the TMSS initialization statement format. Unless there is a gross syntax error, however, these statements are totally ignored.

TMSS Statements in a FOR Group

A number of TMSS statements, because they represent resources that are to be shared, cannot be included within a FOR Group. If any of the following statements is found within a FOR Group, an error message is generated and TMSS processing terminates:

```
FILE CF
FILE SPOOL
FILE VIF
TM USER
```

FOR Group Example

The following is an example of the use of the FOR Group. Only some of the TMSS initialization statements are shown here:

```
...
FILE SPOOL SYS2.TMSPOOL
JAL LOAD SYS2.TMJAL(JAL01)
FOR SMFID(SYS1)
     TM UNIT HANDLE(TAPE1) AS(TAPE)
ENDFOR
FOR SMFID(SYS2,SYS3)
     CPS WRITER LIBRARY ...
ENDFOR
VOL SET V(RESB01,RESB02) RETIRED
...
```

Coding Rules for TMSS Statements

The coding rules for TMSS initialization statements are:

- Physical input records must be 80 bytes long.
- Control statements can begin in any column of the record. Only columns 1 through 72 are scanned.
- Only one definition per control statement.
- Statements exceeding 72 characters can be continued by putting a '+' or '-' as the very last non-blank character of the statement, and continuing the statement on the next input record.
    - The '+' continuation character strips leading blanks on the following statement
    - The '-' continuation character does not.
- Comments start with the characters '/*' and end with the characters '*/'. They can appear anywhere within a statement and can cross input records if the continuation characters '+' or '-' are used as described previously.
- Nested comment delimiters are not allowed. For example:

  ```
  /*  OUTER /* INNER COMMENT */ COMMENT  */
  ```

  is incorrect.
- Strings which include blanks must be enclosed within apostrophes. Blanks outside of enclosed strings delimit parameters.

☞ ***Syntax errors encountered during processing of any control statements cause TMSS to issue a message describing the error(s), and TMSS terminates.***

What follows is a detailed description of all the TMSS initialization statements.

CFM SET

Set Values for Control File

This TMSS initialization statement allows the DEPTH, MINDORM, MAXDORM, MAXHOLD, and MINHOLD values for the Control File to be specified within a FOR group, so that these values can vary between individual systems. The values specified by this statement can override values set by the FILE CF initialization statement. If these values are not set by CFM SET or FILE CF, the defaults are:

- DEPTH defaults to 25 requests.
- MINDORM defaults to 0; that is, requests are processed immediately.
- MAXDORM defaults to 6000, representing one minute.
- MAXHOLD does not have a default value. See Notes.
- MINHOLD defaults to 50, representing one half of a second.

As a starting point, the recommended settings for MINDORM, MAXDORM and MINHOLD are the values in the MASDEF HOLD and DORMANCY parameters on each member for your JES2 checkpoint.

`CFM SET`	`[DEPTH(nnn)] [MINDORM(nnnn)] [MAXDORM(nnnnn)]`
	`[MAXHOLD(nnnnn) | MINHOLD(nnnnn)]`
	`[RESOURCEHOLD(ON | OFF)]`

DEPTH(nnn)

Represents the number of JES2 action requests that can be queued before the Control File Manager waits.

nnn

A value from 0 to 100 requests. The default value is 25. A value of 0 means full synchronization.

MINDORM(nnnn)

Represents the time, expressed in 1/100th of a second, that must elapse before the next request is to be serviced.

nnnn

A value from 0 to 6000, representing up to 1 minute before the next request is to be processed.

MAXDORM(nnnnn)

Represents the maximum time, expressed in 1/100th of a second, that the system waits (when there are no requests) before ThruPut Manager will access the Control File to indicate that it is still active.

nnnnn

A value from 0 to 12000, representing up to 2 minutes. The default value is 6000.

MAXHOLD(nnnnn)

Represents the maximum time, expressed in 1/100th of a second, that the Control File Manager holds the Control File (after the control record has been read).

This keyword is mutually exclusive with MINHOLD.

nnnnn

A value from 50 to 12000, representing up to 2 minutes.

MINHOLD(nnnnn)

Represents the minimum time, expressed in 1/100th of a second, that the Control File Manager holds the Control File (after the control record has been read).

This keyword is mutually exclusive with MAXHOLD.

nnnnn

A value from 0 to 12000, representing up to 2 minutes. The default value is 50.

Unless you have specific requirements, the default value should not be altered.

RESOURCEHOLD(ON | OFF)

Allows you to control the way that JBS and JLS jobs are held.

ON

Indicates that JBS and JLS jobs are to be held by removing their system affinity.

OFF

Indicates that JBS and JLS jobs are to be held using ThruPut Manager holds. This is the default.

Notes:

If more than one source is specified for DEPTH, MINDORM, MAXDORM, MAXHOLD, or MINHOLD, ***the last setting encountered takes effect,*** regardless of whether the source was a CFM SET or FILE CF statement.

MAXHOLD and MINHOLD are mutually exclusive. If neither MAXHOLD nor MINHOLD is specified, the default is MINHOLD(50), or half a second.

If any image of ThruPut Manager must revert to a level without RESOURCEHOLD support, or if JES2 is UNACTIVATED to pre-R4 mode, you must set RESOURCEHOLD to OFF before reverting to the earlier level.

CPS WRITER

Specify CPS Writer Parameters

This statement defines and activates a CPS Writer. One statement is needed for each Writer.

```
CPS WRITER          name
                    [START | DRAIN]
                    [BOTTOMMARGIN(nnn)]
                    [CONSOLE(ucmid | console-name) | ROUTE(nnn)]
                    [DESTINATION(name1 name2 ...)]
                    [PACE(nnnnn)]
                    [PAGELENGTH(nnn)]
                    [TOPMARGIN(nnn)]
```

name

Is a *unique* name for this CPS Writer. It must be 1 to 8 alphabetic, numeric, or national characters long. This parameter is positional and must be specified.

START | DRAIN

START

Indicates that this Writer is active when first defined, or after a COLD start (i.e. the Spool File was formatted). **This is the default status**, and is mutually exclusive with DRAIN.

DRAIN

Indicates that this writer is inactive when first defined, or after a COLD start (i.e. the Spool File was reformatted). This parameter is mutually exclusive with START.

BOTTOMMARGIN(nnn)

Specifies the number of blank lines printed at the bottom of JVLs produced by this Writer.

nnn

nnn must be an integer in the range 0 to 255.

The default value is 0.

CONSOLE(console-name)

Directs output to specific console. This keyword is mutually exclusive with the ROUTE keyword. If it is omitted, the Writer uses the ROUTE keyword or its default.

console-name

Specifies a console name that identifies the console this Writer uses to print Volume Lists.

DESTINATION(name1 name2 ...)

Specifies the destinations to be selected by this Writer.

The default destination is MVL.

(name1 name2 ...)

Specifies one or more Destinations that this Writer will select.

Destinations are assigned to Volume Lists through JAL.

PACE(nnnnn)

Specifies the “pacing factor” for this Writer. This is the approximate number of milliseconds per line of output that should elapse between the time printing begins and the time the next Volume List is selected for this Writer. Pacing protects against a sudden influx of Volume Lists depleting the WQEs.

nnnnn

This operand must be an integer in the range 0 to 30000. The default value of 350 is based on the operating speed of a 3287 dedicated to printing Volume Lists. This keyword only applies when a SPOOL File is used. If no spooling is done, this keyword is ignored.

PAGELENGTH(nnn)

Specifies the page length of Volume Lists printed by this Writer. ***The page length represents the number of lines that can be printed in a page***.

nnn

The value must be an integer in the range 0 to 255. If it is omitted, or is specified as 0, no pagination is performed. When PAGELENGTH is not zero, the Writer inserts blank lines at the end of a Volume List to ensure that the number of lines printed is equal to the PAGELENGTH value.

ROUTE(nnn)

Specifies the MVS route code used for Volume Lists printed by this Writer. This keyword is mutually exclusive with the CONSOLE keyword.

nnn

A valid route code.

The default value is 3.

TOPMARGIN(nnn)

Specifies the number of blank lines printed at the top of Volume Lists.

nnn

It must be an integer in the range 0 to 255.

The default value is 0.

Notes:

More than one CPS Writer can be defined by coding multiple CPS WRITER initialization statements using different Writer names. Specifying the same name more than once is an error.

If a certain type of error occurs, such as an invalid route code or console id, the Writer *will be defined but its initial state will be DRAINed.* This allows you to make corrections using CPS operator commands.

CPS Writers must have names that are unique. Defining duplicate Writers on a single system causes TMSS to terminate with an error message. Duplicate names across systems sharing the same SPOOL File prevent the second CPS Writer from starting, and cause a warning message.

The parameters specified with the CPS WRITER initialization statement define the original status of the Writer. If Volume List Spooling is used, changes made with a CPS WRITER command are checkpointed. The new values are then used in any subsequent startup, until the SPOOL File is reformatted.

If spooling is active when TMSS starts, the status of an existing Writer is determined from the values checkpointed in the SPOOL File.

CPS Writers are owned by the first system that defines them.

DAL LOAD

Load DD Action Language

This statement signals to ThruPut Manager that DAL installation rules are to be activated. It also identifies the dataset that contains the DAL output from the Language Processor. The dataset must be cataloged, and can be either sequential or partitioned.

`DAL LOAD`	`dsname[(membername)]`

`dsname(membername)`

Specifies the name of a cataloged dataset ***containing the internal text*** produced by the Language Processor from your DAL statements.

`membername`

Specifies a PDS member name. This member contains the DAL internal text produced by the Language Processor from your DAL statements.

Notes:

There is no default for this statement.

Multiple DAL LOAD statements for a system are permitted, but each DAL LOAD statement should refer to a different type of DAL.

Different DAL LOAD statements for each type of DAL can be applied to individual systems by using a FOR group.

If you specify multiple DAL LOAD statements for DAL ***of the same type*** for a particular system, ***the last statement encountered is used***.

DAL TRACE

Trace DAL Execution

This statement allows you to activate DAL tracing at TMSS start time. You can either do it for all jobs or for selected jobs.

`DAL TRACE`	`ON [JOBNAME(name1,name2,...)]`

`ON`

Indicates that you want tracing turned on.

`JOBNAME(name1,name2,...)`

This keyword indicates that you want selected job tracing.

`name1,name2,...`

Specifies the name of the jobs for which you want to have the DAL execution traced.

You can specify a complete job name, or you can use the form JOBNAME(ABC*). In this case, any job which has a jobname starting with the letters ABC will be traced.

`STEP#(step)`

This keyword restricts the actions of the statement to a particular step within the jobs being traced. If this keyword is not specified, ***all*** steps are affected. ***This keyword is not valid unless accompanied by the JOBNAME keyword.***

`step`

A step number. Step numbers are relative to the beginning of the job and start at 1.

Note that the specified step is traced for ***all*** jobs being traced.

`ONERROR(ON | OFF)`

Turns DAL tracing ON or OFF for any job that encounters an error. Note that only the first error is traced.

`ON | OFF`

Self-explanatory.

Notes:

If more than one DAL TRACE statement is specified for a system, ***the last statement is effective***. A different DAL TRACE statement can be applied to each system by using a FOR group.

DCS SET

TM/DCS ONLY

SET DCS Parameters

This TMSS initialization statement sets some global defaults for DCS.

```
DCS SET              [ALRTDEST(destination)]
                     [ALRTTIME(time-interval)]
                     [DR( NO | YES)]
                     [AUTOFREE(YES)]
                     [MAXCLAIM(nnn)]
                     [MIMNAME(name-pattern)]
                     [NAG(how-many-times,time-interval)]
                     [SERVICE(CLAIM | CONTEND | STANDBY)]
                     [SNAG(how-many-times,time-interval)]
                     [TRCEDEST(destination)
                     [XCF(NO | YES)]
```

ALRTDEST(destination)

Specifies the default destination for an ALERT. This value is used when a JAL or JECL ALERT request did not specify a destination.

If this keyword is not specified in the initialization statement the default value is DCSALERT.

destination

A valid destination name.

ALRTTIME(time-interval)

Specifies the time, in minutes, after which an ALERT Bulletin is to be written. This value is used when a JAL or JECL ALERT request did not specify a time value.

If this keyword is not specified in the initialization statement the default value is 10 minutes.

time-interval

A value from 1 to 999 representing the number of minutes.

AUTOFREE(YES)

Enables the automatic freeing of datasets that are allocated to the job submitter TSO session.

DR(NO | YES)

Controls dataset reservation processing support by DCS.

`NO`

Indicates that DCS ***should not*** obtain an ENQ during job initiation for the ***absolute dataset name*** associated with a generation dataset, or the ***real dataset name*** associated with an alias. This is the default.

`YES`

Indicates that DCS ***should*** obtain an ENQ during job initiation for the ***absolute dataset name*** associated with a generation dataset, or the ***real dataset name*** associated with an alias. See **Notes** for additional information.

`MAXCLAIM(nnn)`

This parameter specifies the number of jobs that can wait in an initiator in CLAIM mode. ***The value applies to a JES2 node***. If you have multiple JES2 nodes in your DCS complex, then you can assign different values to each node; however, all systems in a JES2 node must have the same value.

If this keyword is not coded, there is no limit to the number of jobs that can wait in initiators.

`nnn`

A value from 1 to 255.

`MIMNAME(name-pattern)`

This parameter is mandatory in MIM environments. It provides the name of MIM's address space to DCS.

`name-pattern`

A 1 to 8 character string representing the name of the started task or job under which MIM runs. In situations where the name varies, you specify a name pattern. For example, you can run MIMSYS1 in one system and MIMSYS2 in a second system. You can simply code MIMSYS* in both systems.

`NAG(how-many-times,time-interval)`

Provides the defaults to NAG TSO holders of datasets needed by a ***batch job.***

`how-many-times`

A value from 0 to 99 that represents the number of times a TSO user is to be nagged. If 0 is specified, it suppresses default nagging.

If this value is not specified, the default is 3 times.

`time-interval`

a value from 1 to 99 that represents the interval, in minutes, between "nags".

If this value is not specified, the default is 5 minutes.

`SERVICE(type)`

Specifies the default dataset service level. If this keyword is not specified, the default is STANDBY.

`type`

Can be CLAIM, CONTEND, or STANDBY.

For an explanation of the meaning of the different service levels refer to ***Dataset Contention Services: System Programming Guide*** Chapter 3. Requesting Dataset Service Levels." Unless you have well understood reasons, we recommend that you use the STANDBY default.

`TRCEDEST(destination)`

Specifies a default destination for all DCS trace statements, inclduing the auto-trace.

`destination`

A valid destination name. If the specified destination is not valid, $DEFAULT is used instead.

`SNAG(how-many-times,time-interval)`

Provides the defaults to NAG TSO holders of datasets needed by a ***started task.***

`how-many-times`

A value from 0 to 99 that represents the number of times a TSO user is to be nagged. If 0 is specified, it suppresses default nagging.

If this value is not specified, the default is 99 times.

`time-interval`

A value from 1 to 99 that represents the interval, in minutes, between "nags".

If this value is not specified, the default is 1 minute.

`XCF(NO | YES)`

Controls the use of XCFM support by DCS.

`NO`

Specifies that DCS is to ***stop*** using XCFM for cross-systems communication, i.e. DCS reverts to using the Control FIle.

`YES`

Specifies that DCS is to ***start*** using XCFM for cross-systems communication.

Notes:

Dataset reservation support does not apply to exempt jobs, unlike other DCS support. Also, the JECL statement /*TM EXEMPT DCS is honored.

DCS dataset reservation support ENQs are always exclusive ***for tape datasets only***.

If TMSS encounters more than one DCS SET statement, ***only the last statement is effective***.

DJC SET

TM/UCS ONLY

SET DJC Parameters

This TMSS initialization statement sets some global defaults for DJC.

`DJC SET`	`[HISTORY_COUNT(n)]`
	`[HISTORY_DAYS(nnn)]`
	`[OUTPUT_IF_NOT_RUN
		PURGE_IF_NOT_RUN]`

`HISTORY_COUNT(n)`

Specifies the default for the number of generations of history to keep for a DJC Group.

`n`

Is a number from the range 0-9, representing the number of generations to keep. A value of 0 indicates that no DJC Group history is to be kept. If this keyword is not specified, the default value is 1.

`HISTORY_DAYS(nnn)`

Specifies the default number of days that a DJC Group history is to be kept. If this keyword is not specified, the default value is 7.

`nnn`

Is a number from the range 1-365, representing the number of days to keep a DJC Group history.

`OUTPUT_IF_NOT_RUN`

Indicates that jobs flushed by DJC are to be placed on the JES2 output queue. This keyword is mutually exclusive with PURGE_IF_NOT_RUN, and is the default.

`PURGE_IF_NOT_RUN`

Indicates that jobs flushed by DJC are to be placed on the JES2 purge queue. This keyword is mutually exclusive with OUTPUT_IF_NOT_RUN.

Notes:

If TMSS encounters more than one DJC SET statement, ***only the last statement is effective***.

ENDFOR

Control Statement

This statement indicates the end of a FOR Group.

`ENDFOR`	

Examples:

```
...
FILE SPOOL SYS2.TMSPOOL
JAL LOAD SYS2.TMJAL(JAL01)
FOR SMFID(SYS1)
       TM UNIT HANDLE(TAPE1) AS(TAPE)
ENDFOR
FOR SMFID(SYS2)
       CPS WRITER LIBRARY ...
ENDFOR
```

Notes:

The ENDFOR statement terminates an FOR Group. It is mandatory if a FOR statement is present.

If another FOR statement is encountered before an ENDFOR, it is considered an error.

The following initialization statements cannot be included within a FOR Group:

```
FILE CF
FILE SPOOL
FILE VIF
TM USER
```

FILE CF

Control File Initialization Statement

This TMSS initialization statement defines and activates the Control File.

☞ ***This is a mandatory file.***
This statement is not allowed in a FOR group.

The DEPTH, MINDORM, MAXDORM, MAXHOLD, and MINHOLD keywords can also be specified using a CFM SET statement. If these values are not set by CFM SET or FILE CF, the defaults are:

- DEPTH defaults to 25 requests.
- MINDORM defaults to 0; that is, requests are processed immediately.
- MAXDORM defaults to 6000, representing one minute.
- MAXHOLD does not have a default. See Notes.
- MINHOLD defaults to 50, representing one half of a second.

The recommended setting for each of the MINDORM, MAXDORM, and MINHOLD values is ***double the JES2 setting used by your installation.*** See the Notes for further discussion of the relationship between the settings for ThruPut Manager and JES2.

`FILE CF`	`dsname [DEPTH(nnn)] [MINDORM(nnnn)] [MAXDORM(nnnnn)]`
	`[MAXHOLD(nnnnn) | MINHOLD(nnnnn)]`

`dsname`

Represents the dataset name of the Control File. It is dynamically allocated. It must be cataloged.

`DEPTH(nnn)`

Represents the number of JES2 action requests that can be queued before the Control File Manager waits.

`nnn`

A value from 0 to 100 requests. A value of 0 means full synchronization.

`MINDORM(nnnn)`

Represents the time, expressed in 1/100th of a second, that must elapse before the next request is to be serviced.

`nnnn`

A value from 0 to 6000, representing up to 1 minute before the next request is to be processed.

`MAXDORM(nnnnn)`

Represents the maximum time, expressed in 1/100th of a second, that the system waits (when there are no requests) before ThruPut Manager will access the Control File to indicate that it is still active.

`nnnnn`

A value from 0 to 12000, representing up to 2 minutes.

`MAXHOLD(nnnnn)`

Represents the maximum time, expressed in 1/100th of a second, that the Control File Manager holds the Control File (after the control record has been read).

This keyword is mutually exclusive with MINHOLD.

`nnnnn`

A value from 50 to 12000, representing up to 2 minutes.

`MINHOLD(nnnnn)`

Represents the time, expressed in 1/100th of a second, that the Control File Manager holds the Control File (after the control record has been read).

This keyword is mutually exclusive with MAXHOLD.

`nnnnn`

A value from 0 to 12000, representing up to 2 minutes.

Unless you have specific requirements, the default value should not be altered.

Notes:

The Control File is a required file. Failure to specify a Control File results in a non-deletable warning message, and all ThruPut Manager components requiring the Control File cannot function. TMSS continues in order to provide a diagnostic environment.

The Control File is automatically formatted upon its first use. Subsequent reformatting can be requested by coding CF=COLD when starting TMSS.

The DEPTH, MINDORM, MAXDORM, MAXHOLD, and MINHOLD values specified by this statement can be overridden by the CFM SET initialization statement. FILE CF can be used to set global values for a complex, then CFM SET can be used within a FOR group to set different values for individual systems.

MAXHOLD and MINHOLD are mutually exclusive. If neither MAXHOLD nor MINHOLD is specified, the default is MINHOLD(50), or half a second.

If FILE CF is specified more than once, ***the last statement encountered is used***.

If more than one source is specified for DEPTH, MINDORM, MAXDORM, MAXHOLD, or MINHOLD, ***the last setting encountered takes effect,*** regardless of whether the source was a CFM SET or FILE CF statement.

FILE CMF

TM/DCS ONLY

DCS Recording File Definition

This statement specifies the name of the file to be used for Contention Management Facility (CMF) dataset contention data recording. This file is a “wrap-around” file that is shared by all systems in a TM/DCS complex. ***This statement is not accepted within a FOR GROUP.***

`FILE CMF`	`dsname`

`dsname`

Represents the dataset name of the DCS recording file. It is dynamically allocated. It must be cataloged.

Notes:

There is no default for this statement.

If this statement is omitted, TM/DCS does not record dataset contention data.

The CMF File is automatically formatted upon its first use. Subsequent reformatting can be requested by coding CMF=COLD when starting TMSS.

If FILE CMF is specified more than once, ***the last statement encountered is used***.

FILE SPOOL

Spool File Definition

This statement specifies that TMSS is to provide spooling services, and provides the name of the SPOOL File. ***This statement is not accepted within a FOR GROUP.***

`FILE SPOOL`	`dsname`

`dsname`

Specifies a cataloged dataset name, pointing to the SPOOL File.

Notes:

There is no default for this statement.

If this statement is omitted and you are using CPS, ThruPut Manager does not perform:

- DCS ALERT spooling.
- CPS Writer status checkpointing.
- DCS ALERTs on a system other than the system on which they were created.

The SPOOL file is automatically formatted upon its first use. Subsequent reformatting can be requested by coding SPOOL=COLD when starting TMSS.

If FILE SPOOL is specified more than once, ***the last statement encountered is used***.

FILE VIF

Define VIF

This statement specifies the name of the dataset used for the Volume Information File. ***This dataset is mandatory***, and must be allocated in a single extent. This dataset must be accessible to any system sharing the SPOOL File *for which CPS Writers are activated*. ***This statement is not accepted within a FOR GROUP.***

`FILE VIF`	`dsname [AUTOFILE(nnn)] [ENTRIES(nnnnn)] [VSEG(nnn)]`

`dsname`

A required positional parameter which specifies a cataloged dataset name for the Volume Information File.

`AUTOFILE(nnn)`

This keyword allows you the specify the time that must elapse before a volume is automatically removed from the VIF.

If you do not code this keyword, the default value is 169 hours. (7 days times 24 hours, plus 1 hour.)

`nnn`

Must be in the range 1 to 999. It represents the number of hours that must elapse before an ***unreferenced*** volume is automatically removed from the VIF.

Every time a reference is made to the volume, the elapsed time is reset to zero.

`ENTRIES(nnnnn)`

Is the maximum number of entries before the VIF becomes full. *This value can be changed only when the VIF is reformatted.*

The keyword ENTRY can also be used here.

`nnnnn`

Must be in the range 1 to 32767.

The default value if this keyword is not present is 7200. (one cylinder of 3380/3390).

Notes:

Volume Information File processing ***must be active*** to support Job Setup Services.

The Volume Information File is a required file. Failure to specify a VIF results in a non-deletable warning message, and Job Setup Services cannot function. TMSS continues in order to provide a diagnostic environment.

If you let the VIF assume the default values, the space required is 1 cylinder of a 3380/3390 DASD device. Cylinder allocation improves the efficiency of the VIF channel programs.

The VIF is automatically formatted upon its first use. Subsequent reformatting can be requested by coding VIF=COLD when starting TMSS.

Once the VIF has been formatted, it cannot be moved to an unlike device (e.g. from an IBM 3380 to an IBM 3390). If this occurs, TMSS requests permission to reformat the file. The existing VIF cannot be used on the new device type.

If more than one FILE VIF statement is specified, ***the last statement encountered is used***.

FOR

Control Statement

This statement indicates the beginning of a FOR Group. If The SMFID of the system processing the statement matches one of the SMFIDs in this statement, then the initialization statements following the FOR statement are processed. Otherwise, they are bypassed.

`FOR`	`SMFID(smf1[,smf2,...])`

`SMFID(smf1[,smf2,...])`

Specifies one or more SMFIDs to be matched.

`smf1[,smf2,...]`

Represent valid SMFIDs.

Examples:

```
...
FILE SPOOL SYS2.TMSPOOL
JAL LOAD SYS2.TMJAL(JAL01)
FOR SMFID(SYS1)
    TM UNIT HANDLE(TAPE1) AS(TAPE)
ENDFOR
FOR SMFID(SYS2,SYS3)
    CPS WRITER LIBRARY ...
ENDFOR
VOL SET V(RESB01,RESB02) RETIRED
...
```

Notes:

The FOR statement marks the beginning of a FOR group. It must be terminated with an ENDFOR statement.

If another FOR statement is encountered before an ENDFOR, it is considered an error.

The following initialization statements cannot be included within a FOR Group:

```
FILE CF
FILE SPOOL
FILE VIF
TM USER
```

JAL LOAD

Load Job Action Language

This statement signals to ThruPut Manager that installation rules are to be activated. It also identifies the dataset which contains the output from the Language Processor. The dataset must be cataloged, and can be either sequential or partitioned.

`JAL LOAD`	`dsname[(member-name)]`

`dsname(member-name)`

Specifies the name of a cataloged dataset ***containing the internal text*** produced by the Language Processor from your JAL statements.

`member-name`

Specifies a PDS member name. This member contains the JAL internal text produced by the Language Processor from your JAL statements.

Notes:

There is no default for this statement.

If no JAL internal text is provided, ThruPut Manager provides JCL checking only. The jobs are requeued with the same attributes they had when they were submitted.

If multiple JAL LOAD statements are specified for a system, ***the last statement encountered is used***. A different JAL LOAD statement can be applied to each system by using a FOR group.

JAL TRACE

Trace JAL Execution

This statement allows you to activate JAL tracing at TMSS start time. You can either do it for all jobs or for selected jobs.

`JAL TRACE`	`ON [JOBNAME(name1,name2,...)]`

`ON`

Indicates that you want tracing on.

`JOBNAME`

This keyword indicates that you want selected job tracing.

`name1,name2,...`

Specifies the name of the jobs you want to have the JAL execution traced.

You can specify a complete job name, or you can use the form JOBNAME(ABC*). In this case, any job which has a jobname starting with the letters ABC will be traced.

Notes:

If more than one JAL TRACE statement is specified for a system, ***the last statement is effective***. A different JAL TRACE statement can be applied to each system by using a FOR group.

JSS RECALL

Alter the Security Environment for DFSMShsm Recalls

This statement allows you to alter the security environment under which DFSMShsm recalls are processed.

`JSS RECALL`	`[USER	TM]`

`USER`

Indicates the security environment associated with the batch job which triggered the recalls.

`TM`

Indicates the security environment associated with the ThruPut Manager started task.
This is the default.

Notes:

The recalls affected are those initiated implicitly as a result of the job being managed by TM/SLM or explicitly as a result of a "JSS RECALL MAX_WAIT" JAL directive. It does not affect those initiated as a result of a "JSS RECALL DELAY_FOR" JAL directive. DELAY_FOR recalls are always performed under the security environment associated with the batch job which triggered the recalls.

JSS SET

Bypass Allocation Processing for IEFBR14-type Programs

This statement allows you to specify a list of programs for which certain types of allocation processing should not be performed, such as IEFBR14. Allocation processing will be bypassed under these circumstances:

- If the primary disposition for a dataset allocated by this type of program is DELETE, HSM recalls are not necessary.
- If the dataset is a tape dataset, it will never be opened, and allocation is not needed.

```
JSS SET          PROGRAM_NAMES(program1[,program2,...]) [BYPASS_RECALLS]
                                                        [BYPASS_TAPE_ALLOCATION]
```

PROGRAM_NAMES(program1[,program2,...])

Indicates that you want to bypass allocation processing for the listed programs.

program1[,program2,...]

Is a list of one or more program names for which allocation processing is to be bypassed.

BYPASS_RECALLS

Specifies that bypassing allocation processing for HSM datasets with a primary disposition of DELETE is to apply to the listed programs.

BYPASS_TAPE_ALLOCATION

Specifies that bypassing tape allocation applies to the listed programs.

Notes:

Multiple JSS SET statements ***for different program names*** are cumulative.

If multiple JSS SET statements are specified ***for the same program name***, the last one encountered takes effect. This means that if both BYPASS_RECALLS and BYPASS_TAPE_ALLOCATION are desired, they should be specified on the same JSS SET statement.

If the primary disposition is DELETE, the dataset is deleted even though allocation processing is bypassed.

DAL is invoked for datasets in steps that execute a listed program, even if all datasets are bypassed.

Bypassed datasets are ***not*** reflected in the counts shown by the JAL Descriptors $HRECALL, $HRECALL_DASD, and $HRECALL_TAPE.

JTS OPTIONS

TM/UCS ONLY

Set Defaults

Description:

This TMSS initialization statement allows you to change the defaults provided with TM/UCS for JTS. It is optional.

```
JTS OPTIONS          [AFTERNOON(hhmm)]
                     [DEFTIME(hhmm)]
                     [MAXDAYS(nnn)]
                     [MAXFAIL | MAXWARN]
                     [MORNING(hhmm)]
                     [OLDFAIL | OLDWARN]
                     [TONIGHT(hhmm)]
```

AFTERNOON

This keyword allows you to alter the default value for AFTERNOON, which is 1300. The value is substituted for the AFTERNOON keyword on the JECL statement /*JTS HOLD_UNTIL.

hhmm

The value to be substituted for 1300.

DEFTIME

This keyword allows you to alter the default for the job “hold until“ time when a /*JTS HOLD_UNTIL statement does not have a TIME value. The default value is 0800.

hhmm

The value to be substituted for 0800.

MAXDAYS

The maximum number of days in the future that a job can be delayed by JTS. The default value for JTS is 60.

nnn

The number of days to be substituted for the value 60. The minimum value is 1, the maximum value is 366.

MAXFAIL

Requests that JTS ***fail*** jobs that exceed the maximum number of days (MAXDAYS value). This mode is the default.

This keyword is mutually exclusive with MAXWARN.

MAXWARN

Requests that JTS ***warn*** jobs that exceed the maximum number of days (MAXDAYS value). A warning message is generated and the /*JTS HOLD_UNTIL statement is ignored. JTS will not manage these jobs.

This keyword is mutually exclusive with MAXFAIL.

MORNING

This keyword allows you to modify the default value for MORNING, which is 0800. The value is substituted for the MORNING keyword on the JECL statement /*JTS HOLD_UNTIL.

hhmm

The value to be substituted for 0800.

OLDFAIL

Requests that JTS ***fail*** jobs that specify a historical (past) date.

This keyword is mutually exclusive with OLDWARN.

OLDWARN

Requests that JTS ***warn*** jobs that specify a historical (past) date. A warning message is generated and the /*JTS HOLD_UNTIL statement is ignored. JTS will not manage these jobs.

This keyword is mutually exclusive with OLDFAIL.

TONIGHT

This keyword allows you to alter the default value for TONIGHT, which is 1830.

This value is substituted for the TONIGHT or NIGHT keyword on the JECL statement /*JTS HOLD_UNTIL.

hhmm

The value to be substituted for 1830.

SPS DEFINE

SYSOUT Printing Services Options

This statement defines a set of options to be used for output printed using JES2 SYSOUT.

```
SPS DEFINE          name
                    CLASS(class)
                    [COPIES(nnn)]
                    [FORM(form)]
                    [HOLD]
                    [JESDESTINATION(dest)]
                    [BOTTOMMARGIN(nnn)]
                    [LEFTMARGIN(nnn)]
                    [PAGELENGTH(nnn)]
                    [TOPMARGIN(nnn)]
```

name

Is a *unique* name for this definition of SYSOUT characteristics. It must be 1 to 8 alphabetic, numeric, or national characters long. This parameter is positional and must be specified.

CLASS(class)

Specifies the output class for this SYSOUT definition. This keyword is required.

class

Is a valid JES2 output class.

COPIES(nnn)

Specifies how many copies of the output to print. If this keyword is not specified, SPS generates a single copy.

nnn

Is a number in the range 1 to 999.

FORM(form)

Specifies the name for special forms on which to print output.

form

Is a one to four character forms name.

HOLD

Specifies that the output is to be held after SPS processing and before it is sent to the printer.

`JESDESTINATION(dest)`

Specifies a JES2 destination to which output is sent.

`dest`

Can be a remote or local terminal, a node, a node and remote workstation, a local device or group of devices, or a node and userid.

`BOTTOMMARGIN(nnn)`

Specifies the number of blank lines printed at the bottom of output printed using this SYSOUT definition.

`nnn`

nnn must be an integer in the range 0 to 255.

The default value is 0.

`LEFTMARGIN(nnn)`

Specifies the number of blanks inserted at the left of each line printed using this SYSOUT definition.

`nnn`

nnn must be an integer in the range 0 to 60.

The default value is 0.

`PAGELENGTH(nnn)`

Specifies the page length of output printed using this SYSOUT definition. ***The page length represents the number of lines that can be printed on a page***.

`nnn`

The value must be an integer in the range 0 to 255. If it is omitted, or is specified as 0, no pagination is performed. When PAGELENGTH is not zero, SPS inserts blank lines at the end of the output to ensure that the number of lines printed is equal to the PAGELENGTH value.

`TOPMARGIN(nnn)`

Specifies the number of blank lines printed at the top of the output.

`nnn`

The value must be an integer in the range 0 to 255.

The default value is 0.

Notes:

Multiple sets of SPS output options can be defined by coding multiple SPS DEFINE statements using different names. Specifying the same SPS name more than once is an error.

In an environment using only SPS, output without a valid destination is queued in SYSOUT class A and is held. For a description of how such output is handled in an environment that also uses CPS refer to “Chapter 4. Printing Services.”

TBL LOAD

Load Table for DAL or JAL

This statement specifies that TMSS is to load a table to be used in DAL or JAL.

`TBL LOAD`	`[n | SAC] dsname[(member-name)]`

`n`

A value from 1 to 9, indicating which table is to be loaded.

`SAC`

Indicates that you want to load the Software Access Control (SAC) table, which links specific program product and commands to JBS Binding Agents.

`dsname(member-name)`

Specifies the name of the dataset where the table is to be loaded from.

`member-name`

The name of the member if the table resides in a PDS.

Notes:

Multiple tables can be loaded by coding multiple TBL LOAD statements specifying different table identifiers. More than one TBL LOAD statement for the same table identifier is an error.

The SAC table is only available with TM/JBS.

TM ATL

Define ATL Unit Names

This statement is used to indicate the esoteric unit name(s) associated with the Memorex (Sutmyn) 5400 ATL.

```
TM ATL              [AUTOMATED(unitname-list)]
                    [MANUAL(unitname-list)]
                    [VTL(unitname-list)]
                    [3480(NO)]
                    [3490(NO)]
                    [3590(NO)]
```

AUTOMATED(unitname-list)

This keyword defines to ThruPut Manager the esoteric unit names that ***must always be served*** by the Memorex 5400 ATL, regardless of where the volume is located.

unitname-list

Represents a list of one or more actual esoteric unit names, separated by commas.

MANUAL(unitname-list)

This keyword indicates to ThruPut Manager the esoteric unit names that are to be served manually regardless of where the volume is located.

unitname-list

Represents a list of one or more actual esoteric unit names, separated by commas.

VTL(unitname-list)

This keyword indicates to ThruPut Manager the esoteric unit names that are to be used to direct allocation to the Virtual Tape Library.

unitname-list

Represents a list of one or more actual esoteric unit names, separated by commas.

3480(NO)

Indicates that 3480 units are not to be considered automated, that is, there are no 3480 units serviced by the automated library.

3490(NO)

Indicates that 3490E units are not to be considered automated, that is, there are no 3490E units serviced by the automated library.

`3590(NO)`

Indicates that 3590 units are not to be considered automated, that is, there are no 3590 units serviced by the automated library.

Notes:

Multiple TM ATL statements are cumulative.

TM BATCH

Submit a Batch of Commands

This statement is used to submit a batch of ThruPut Manager commands at TMSS initialization time. Note that the commands are processed after TMSS initialization is completed. That is, this initialization statement simply queues all the commands for later execution. You can have more than one ‘TM BATCH’ command in your initialization stream.

The rules for storing the text representing the commands are as follows:

- Only columns 1 to 72 inclusive are processed.
- Records containing /* starting in column 1 are treated as comments.
- Commands must be contained within a single statement. That is, continuation to another statement is not allowed.
- The response from a command is sent to the master console, unless the L= keyword was specified on the command.

TM BATCH	**dataset-name[(member-name)] [DISPLAY]**

dataset-name(member-name)

The name of the dataset where the commands to be submitted are stored. It can be a sequential or partitioned dataset.

member-name

If the dataset is partitioned, then the member name must be specified.

DISPLAY

If this keyword is specified, the commands are displayed before they are executed.

Notes:

There is no limit on the number of TM BATCH statements.

TM BTLS

Define BTLS Unit Names for VTS

This statement is used to indicate the esoteric unit name(s) and library names used for VTS by the IBM 3495 robotic tape library running in Basic Tape Library System (BTLS) mode.

`TM BTLS`	`VTS_UNIT(unitname-list) VTS_LIBRARY(library-list)`

`VTS_UNIT(unitname-list)`

This keyword defines to ThruPut Manager the esoteric unit names that are used for VTS by the IBM 3495 robotic tape library running in BTLS mode.

`unitname-list`

Represents a list of one or more actual esoteric unit names, separated by commas.

`VTS_LIBRARY(unitname-list)`

This keyword indicates to ThruPut Manager the library names that are used for VTS by the IBM 3495 robotic tape library running in BTLS mode.

`library-list`

Represents a list of one or more library names, separated by commas.

Notes:

Multiple TM BTLS statements are cumulative.

TM CATALOG SET

Define ThruPut Manager Catalog Search Options

This statement specifies the options used by ThruPut Manager when searching catalogs.

TM CATALOG SET	TIMEOUT[(time)] [RQ_PRIORITY(prio)] NO_TIMEOUT

NO_TIMEOUT

Indicates that the Job Analyzer should ***not*** terminate analysis of the job when MVS Catalog Management does not respond.

This keyword is mutually exclusive with TIMEOUT and RQ_PRIORITY

TIMEOUT[(time)]

Indicates that the Job Analyzer should terminate analysis of the job when MVS Catalog Management has not responded within the time allowed.

This keyword is mutually exclusive with NO_TIMEOUT.

time

Is from the range 1-99, and specifies the number of minutes the Job Analyzer will wait before terminating a job.

The default value for TIMEOUT is 5 minutes.

RQ_PRIORITY(prio)

Indicates that a job terminated because MVS Catalog Management failed to respond should be assigned a specific JES2 priority when requeued.

If this keyword is omitted, the default requeue priority is 0.

prio

Is a number from the range 1-15, specifying the JES2 priority to be assigned.

Notes:

The RQ_PRIORITY keyword is useful for preventing the job from being selected immediately after it is requeued and before other jobs have had a chance to be selected.

TM DLM

Defined DLM Unit Names

This statement defines the device characteristics and tape management library that are associated with this pool of devices.

```
TM DLM              [ESOTERIC(unitname-list)]
                    [STORGRP(storgrp-list)]
                    [LIBRARY(library-list)]
                    [DEVTYPE(3480|3490|3590)]
```

ESOTERIC(unitname-list)

This keyword defines to ThruPut Manager the esoteric unit names that ***must always be served*** by the EMC:DLM, regardless of where the volume is located.

unitname-list

Represents a list of one or more actual esoteric unit names, separated by commas.

STORGRP(unitname-list)

This keyword indicates to ThruPut Manager the Storage Groups that are to be for the new EMC:DLM managed volumes.

storgrp-list

Represents a list of one or more actual SMS storage groups, separated by commas.

LIBRARY(library-list)

This keyword indicates to ThruPut Manager the IBM tape library names that are to be used to detect existing EMC:DLM volumes.

library-list

Represents a list of one or more actual IBM tape library names, separated by commas.

DEVTYPE

This keyword lets you specify the device type to be used in resource classification.

Notes:

Multiple TM DLM statements are cumulative.

TM EXIT

Define an Installation Exit

This statement specifies which installation exits are to receive control, and their initial status.

`TM EXIT`	`nn module-name [ACTIVE | INACTIVE | PERMACTIVE]`
	`[DATA(character-string)]`
	`[TRACE | NOTRACE]`

`nn`

Is an exit number, from 1 to 19, 21, 24, or 31. See the manual ***Exits: System Programming Guide*** for detailed descriptions of these exits.

`module-name`

Specifies the name of the module to be given control at the exit point indicated by ***nn***.

`ACTIVE`

Specifies that the exit is loaded and its initial state is active.

This is the default.

`INACTIVE`

Specifies that the exit is loaded but its initial state is inactive. It remains inactive until activated through an operator command.

`PERMACTIVE`

Specifies that the exit is loaded and its state is permanently active. It cannot be deactivated through the use of the TM EXIT operator command.

`DATA(character-string)`

This keyword allows you to specify a character string that is passed to the exit at exit invocation time.

`character-string`

A character string up to 255 characters long. If it includes special characters or blanks, it must be enclosed within single apostrophes.

`TRACE`

Sets the initial status of this exit as being eligible for tracing.

`NOTRACE`

Sets the initial status of this exit as not being eligible for tracing.

This is the default.

Notes:

The same module name can be specified for more than one exit point.

Specifying a TM EXIT statement more than once for the same exit number causes an error.

Because installation exits associated with the Job Analyzer can run in any active initiator address space, the TM EXIT statement processor enforces the requirement that Job Analyzer exits must come from LPA or a LINKLIST dataset.

The TRACE/NOTRACE keywords of the TM EXIT statement determine the eligibility of the exit for tracing. No tracing is actually done unless there is also a TM TRACE initialization statement, or a TM TRACE command is issued. See the description of the TM TRACE statement below, or the TM TRACE operator command in the ***Operating Guide***.

The TRACE/NOTRACE keywords establish the *initial* status of the exit's trace eligibility. This status can be altered after TMSS is started by using the TM EXIT command.

TM SMF

Define SMF Data Collection Parameters

This statement specifies which SMF user record is used to write SMF data, as well as which data are to be collected. For a description of how to implement SMF data collection, see "Chapter 10. SMF Data Collection for Job Analysis."

```
TM SMF          TYPE(smf-record) [ANALYZER | ANALYZER(table-id[,subtable-id])][DBS][DCS][SLM]
```

`TYPE(smf-record)`

Indicates which SMF record is used.

`smf-record`

Is a number from the range 128-255.

`ANALYZER`

Indicates that the Job Analyzer is to collect the default SMF data for the Job Analyzer.

`ANALYZER(table-id[,subtable-id])`

Indicates that the Job Analyzer is to collect additional SMF data for the Job Analyzer. See **Notes** below.

`table-id`

Is a number from the range 1-9, indicating the ThruPut Manager table that contains the token names for data to be collected.

`subtable-id`

Is a 1-24 character name identifying the portion of the ThruPut Manager table that contains the token names for data to be collected.

`DBS`

Indicates that the DBS application is to collect SMF data. Refer to the ***Drive Booking Services: System Programming Guide***.

`DCS`

Indicates that the DCS application is to collect SMF data. Refer to the ***Dataset Contention Services: System Programming Guide***.

`SLM`

Indicates that SLM data is to be recorded. For further details, refer to the ***Setup Guide to ThruPut Manager AE Essentials***.

Notes:

By default, ThruPut Manager collects certain information. To collect data other than the default, you must create a ThruPut Manager table specifying which data are to be collected. For further information, refer to "Chapter 10. SMF Data Collection for Job Analysis."

TM TRACE

Enable/Disable Tracing

This statement activates and deactivates the ThruPut Manager tracing facility. The options described here allow you to trace installation exit activity. The trace facility can also be used to gather data for problem reporting, but these options should only be activated on the explicit instructions of Customer Support, therefore they are not documented here. If there is a need for this capability, Customer Support will provide the specific commands you should use.

```
TM TRACE            ON EXITS
```

`ON EXITS`

Required keywords to enable tracing of ThruPut Manager installation exits.

Notes:

Tracing for Job Analyzer exits (1 through 9, and 19) goes to the job's SYSMSGS dataset.

Tracing for TMSS exits (10 through 18, 30, and 31) goes to GTF; therefore, GTF must be active and the GTF trace options include tracing for USR event number 249.

An installation exit must be made eligible for tracing, either through the TM EXIT statement or the TM EXIT command. See the TM EXIT statement description in this chapter, or the TM EXIT command description in the ***Operating Guide***.

If exit tracing is active for an exit when analysis of a job starts, all calls to that exit for that job are traced, even if an operator command disables tracing before the analysis is finished.

If more than one TM TRACE statement is specified for a system, ***the last statement is effective***. A different TM TRACE statement can be applied to each system by using a FOR group.

TM UNIT HANDLE

Define Unit Name Mapping

This statement allows you to inform ThruPut Manager of unit name changes that occur after the job has been processed by ThruPut Manager. This facility handles situations where DASD poolers or installation exits alter unit names dynamically.

`TM UNIT HANDLE`	`UNIT(unit-name1) AS(unit-name2)`

`UNIT(unit-name1)`

Identifies the unit name to be altered.

`unit-name1`

The unit name as coded in JCL. More accurately, this is the unit name as seen by ThruPut Manager.

`AS(unit-name2)`

Identifies the “new” unit name for Job Analysis purposes.

`unit-name1`

The “new” unit name.

Notes:

☞ ***The actual JCL unit names are not altered. This is only for ThruPut Manager job analysis, so proper resource classing can take place.***

TM UNIT HANDLE statements are cumulative.

If more than one TM UNIT HANDLE statement is specified for a particular unit name, ***the last statement is effective***.

TM UNIT SET

Set Default Unit Name

This statement establishes a default unit name to be used when a unit name is required but has not been provided in JCL.

`TM UNIT SET`	`DEFAULT(unit-name1)`

`DEFAULT(unit-name1)`

Specifies the unit name that is to be used when no unit name is provided in JCL.

`unitname`

Is the default unit name, which must be defined to the system.

Notes:

☞ ***The actual JCL is not altered. This default is for ThruPut Manager Job Analysis only, so that proper resource classing can take place.***

If more than one TM UNIT SET statement is specified, ***the last statement is effective***.

TM USER

Define User Areas

This statement defines user areas that can be associated with each job. Space is then provided for you to place data. ***This statement is not accepted within a FOR GROUP.***

`TM USER`	`[JOB(nnn)]`
	`[VOLUME(nn)]`

`JOB(nnn)`

Indicates that you want a user area allocated to each job.

The default value is 100.

`nnn`

Represents the number of bytes to be allocated. It can be from 1 to 255.

`VOLUME(nn)`

Indicates that you want a user area allocated to each volume that is referenced by a job.

The default value is 12.

`nn`

Represents the number of bytes to be allocated. It can be from 1 to 99.

Notes:

If you are using the SPOOL File, changing this statement will require a SPOOL cold start.

If more than one TM USER statement is specified, ***the last one is effective***.

TM VTAPE

Define DFSMS DATACLAS for CA-Vtape

This statement is used to indicate the DFSMS DATACLAS used for virtual tape volumes by CA-Vtape (formerly known as SAMS:VTAPE) support.

`TM VTAPE`	`DATACLAS(vtape_dataclas[,...])`
	`[INCLUDE_ON_VL(AUTOMATED | MANUAL)]`

`DATACLAS(vtape_dataclas[,...])`

This keyword defines to ThruPut Manager the DFSMS DATACLAS used for virtual tape volumes by CA-Vtape support.

`vtape_dataclas`

Represents a list of one or more DATACLAS names, separated by commas.

`INCLUDE_ON_VL(AUTOMATED | MANUAL)`

This keyword indicates that you want the job's profile to include the physical volumes used to stack the virtual volumes.

`AUTOMATED`

Indicates the volumes are to be included and treated as automated (robotic) volumes.

`MANUAL`

Indicates the volumes are to be included and treated as manual volumes.

Notes:

If you specify multiple TM VTAPE statements:

- The DATACLAS names you specify are cumulative.
- The setting for INCLUDE_ON_VL, if specified, is determined by the last occurrence encountered.

TM VTFM

Define VTFM Parameters

This statement is used to indicate the esoteric and generic unitnames, and the subsystem name needed for VTFM support.

`TM VTFM`	`ESOTERIC(e-unitname[,...]) GENERIC(g-unitname[,...]) SSNAME(subsystem)`

`ESOTERIC(e-unitname[,...])`

This keyword defines to ThruPut Manager the esoteric unitname used for virtual tape volumes by VTFM support.

`e-unitname[,...]`

Represents a list of one or more esoteric unitnames, separated by commas.

`GENERIC(g-unitname[,...])`

This keyword defines to ThruPut Manager the generic unitname used for virtual tape volumes by VTFM support.

`g-unitname[,...]`

Represents a list of one or more generic unitnames, separated by commas.

`SSNAME(subsystem)`

This keyword defines to ThruPut Manager the subsystem name used for virtual tape volumes by VTFM support.

`subsystem`

Represents a valid susbsystem name.

Notes:

One TM VTFM statement is required for each VTFM sybsystem.

Multiple TM VTFM statements are cumulative.

UCS SET

TM/UCS ONLY

Adjust Accumulated Queue Time for UCS Held Jobs

UCS allows users to place jobs in either MHS_USER or JTS hold. Since these jobs cannot run until released, the accumulated queue time does not represent a delay in service. The UCS SET initialization statement provides a way to adjust the accumulated queue time for non-SLM jobs that have been placed in MHS_USER or JTS hold. Assuming that a numeric value is specified:

- If the job has not accumulated at least the specified number of hours in the queue, there is no adjustment.
- Otherwise, the specified number of hours is subtracted from the release time, resulting in an apparent more recent arrival time.

```
UCS SET                  [nn | NOW | OFF]
```

`nn`

Is a number ranging from 1-24, indicating the number of hours ***to be subtracted from the release time*** to determine the new apparent arrival time for a non-SLM job released from MHS_USER or JTS hold.

`NOW`

Indicates that for a non-SLM job just released from MHS_USER or JTS hold, the arrival time should be reset to the current time.

`OFF`

Indicates that a non-SLM job just released from MHS_USER or JTS hold should retain its original arrival time.

Examples:

Assume JOBA arrives at 8:00 AM but is placed in MHS_USER hold until it is released at 4:00 PM. The initialization statement is:

```
UCS SET 3
```

Because the job has accumulated eight hours of queue time, the value from the UCS SET statement is applied. The release time is decreased by three hours so that the arrival time appears to be 1:00 PM.

Assume JOBB also arrives at 8:00 AM but is released at 10:00 AM. Because less than three hours have elapsed, there is no adjustment.

Notes:

This initialization statement affects only non-SLM jobs that have been released from MHS_USER or JTS hold.

If there is no UCS SET initialization statement and the UCS SET command has not been issued, there is no adjustment.

When a job is released, TM/UCS applies the UCS SET value specified for the ***system*** where the release occurred. Since this could result in asymmetric behavior, we recommend that the UCS SET statement ***not*** be included in a FOR Group.

UDF SET

Set UDF JES2 Job Number Heading

This TMSS initialization statement sets the column heading that UDF recognizes for the column containing JES2 job numbers.

`UDF SET`	`manager_JOBID(header1[,...,headern])`

`manager_JOBID(header1[,...,headern])`

Specifies the headers that UDF recognizes for the JES2 job number column in UDF displays.

`manager`

Specifies the spool manager in use at your installation, and can be one of:

`IOF`

Indicates IOF is the spool manager.

`JESM`

Indicates JES Master is the spool manager.

`SDSF`

Indicates SDSF is the spool manager.

`SYSV`

Indicates CA/Sysview is the spool manager.

`header1[,...,headern]`

Is one or more strings of 1 to 12 characters specifying the heading(s) that the spool manager can use for the JES2 job number column. If the string contains a blank, for example ‘JOB ID’, it must be enclosed in apostrophes. Specifying more than one string allows TM/UDF to recognize the JES2 job number column in cases where the heading could vary.

Examples:

```
UDF SET IOF_JOBID('JES2 NO.')
```

Here the spool manager is IOF. This statement indicates that the installation has altered the JES2 job number column heading to read JES2 NO.

```
UDF SET SYSV_JOBID(JES2#,'JOB NUMBERS')
```

In this example, the spool manager is CA/Sysview, and TM/UDF is instructed to recognize either JES2# or JOB NUMBERS as the heading for the JES2 job number column.

Notes:

The specified headers are recognized ***in addition to*** the default headers for the spool manager.

It is valid to specify UDF SET statement for more than one spool manager.

Multiple UDF SET statements for the same spool manager are cumulative.

VOL SET

Set Status of Volumes in DASD Volume Table

This initialization statement sets the status of a given DASD volume serial number in the ThruPut Manager DASD Volume Table. This table is used by ThruPut Manager during the Job Analysis process.

`VOL SET`	`V(volser1,volser2,...) status`

`V(volser1,volser2,...)`

Indicates that a list of DASD volume serial numbers is to be given the status indicated by the keyword following.

`volser1,volser2,...`

Is a list of volume serial numbers. In this list, the wildcard character ‘*’ can be used as part of the volser. To be treated as a wildcard, ***it must be the last character***. When ‘*’ is coded, any volume reference that begins with the characters preceding the ‘*’ results in a match. There must be ***at least one character*** preceding the ‘*’.

If the actual volume serial number you want to code has a trailing asterisk, then you must enclose the volume serial number in apostrophes.

`status`

Is the status to be assigned to the volume, and can be one of:

`IGNORE`

Specifies that when this volume entry is coded as a single VOL=SER entry in JCL, it is to be ignored. This replicates the behavior of the Dummy Storage Group in SMS.

`MIGRATE`

Sets the volume entry as the volume serial number used by your storage management system to detect files that have been migrated. For example, ARCIVE is used for DMS/OS environments.

`RESIDENT`

Sets the volume entry as resident, regardless of physical availability. Such volumes are also treated as already mounted.

`RETIRED`

Sets the volume entry to indicate that a previously existing online volume has been “retired”. Jobs with catalog entries or hard coded values referring to this type of volume are failed with an informative message.

UNAVAILABLE

Sets the volume entry as unavailable for use by jobs. Jobs requesting this type of volume are not eligible to be selected for analysis or execution on this system. Once the volume becomes available, the jobs should be released for normal ThruPut Manager processing.

Examples:

```
VOL SET V(RES*,MAS*) RETIRED
```

In this example any volume that starts with ‘RES’ or ‘MAS’ results in a match.

```
VOL SET V('RES*',MAS*) RETIRED
```

In this case, the first volume serial number in the list is enclosed in apostrophes; therefore is not treated as a wildcard character. For the second element in the list, it is treated as a wildcard character.

Notes:

More than one VOL SET statement can be specified for a system. The effects of multiple VOL SET statements for the same system are cumulative. Different VOL SET statements can be applied to each system by using a FOR group.

When more than one entry in the list results in a match, the most specific entry is used. For example, if your list contains the following entries:

```
RES*   RETIRED
RESB*  UNAVAILABLE
```

A volume with a serial number of RESB40 will be treated as UNAVAILABLE because RESB* is more specific than RES*.

ThruPut Manager automatically detects volumes migrated by DFHSM, therefore you do not need to a VOL SET statement for this environment. ThruPut Manager assumes that the volume serial for the migrated volume will be MIGRAT, which is the default defined by IBM.

Chapter 3. File Definition Services (FDS) Function

This chapter describes planning considerations for the ThruPut Manager Files. The space and allocation techniques are described.

Introduction

This chapter provides all the information required to implement the supporting files that ThruPut Manager requires. The chapters describing the ThruPut Manager Functions indicate which files they require, if any.

File to Function Cross Reference

This table shows the relationship between ThruPut Manager functions and ThruPut Manager files.

Cross Reference	
File	**Function**
CF	Dataset Contention Services.
	Job Setup Services.
	Job Binding Services.
	Job Chaining Services.
	Job Classing Services.
	Job Limiting Services.
	Mellon Compatibility Services.
	Multi-hold Services.
SPOOL	Console Printing Services (CPS) Writer. This file is not required if hard copy Volume Lists are not printed and you are not using CPS to handle DCS Alerts.

Cross Reference	
File	**Function**
CMF	Dataset Contention Services Reporting. This file is not required if DCS Reporting is not used.
AF	Drive Booking Services (DBS) Function.

The Control File

Job Setup Services and the optional components Job Binding Services, Job Limiting Services, Job Chaining Services, and Dataset Contention Services require the ThruPut Manager Control File to be active. Note that the Control File is a critical resource. Access times to this file can impact the overall performance of the system.

Implementation Summary

The steps for implementing the Control File are:

- Find a device that delivers good to excellent service times.
- Allocate the file with 64 cylinders of 3380 or 3390 space.
  - If you are using TM/DCS across multiple JESplexs, allocate one Control File per JESplex.
- Add the FILE CF statement to the TMSS initialization statements.
  - If you are using TM/DCS, add the XCF(ON) keyword to the DCS SET initialization statement.
- Start TMSS using the COLD parameter for CF.
- Confirm the TMSS formatting request by replying ‘Y’.

FILE CF Initialization Statement

The FILE CF statement must be included in your TMSS initialization statements. The format of this statement is documented in “Chapter 2. TMSS Initialization Statements .”

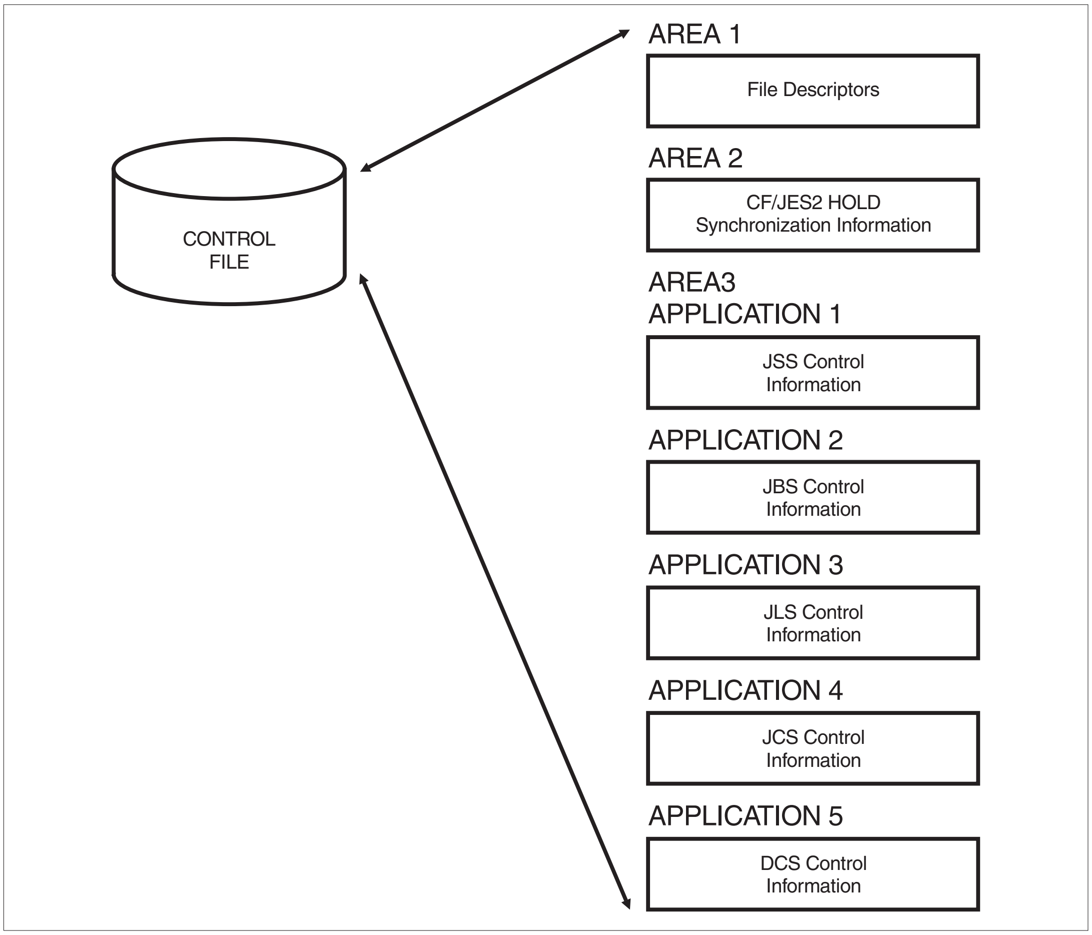
Control File Structure

For the JSS application, the Control File contains information necessary to manage the JSS Hold Extensions. For applications such as JBS and JLS, the Binding Agents, Limiting Agents, and all the necessary information to manage the jobs dependent on these Agents are stored in the Control File.

The conceptual structure of the Control File is simple. It is organized into three distinct but related areas. Here is a schematic representation of the file:

The first area contains information describing the file. For example, file size, number of applications, parameters such as MAXDORM, and other items required to manage the file are stored in this area.

The second area maintains all the information required to coordinate the relationship between ThruPut Manager held jobs and JES2. All the necessary infor-



Control File Structure

mation to synchronize JES2 and ThruPut Manager is contained here. The link between JES2 and the specific application is also contained in here.

The third area is where the application-specific information is kept. There is one section per application.

Each application section contains all the detail information to control the jobs receiving its services. For example, in the case of JSS the information needed to hold and release jobs automatically is contained in the JSS application section. For JBS, the Agents that have been defined are stored in the JBS application section, and as the relationships between jobs and Agents evolve, the necessary connections are also kept here. Similar relationships exist for each application using the Control File.

☞ ***The Control File is a high performance file and should be placed in a device that can deliver good to excellent service times. It requires 64 cylinders of a 3380 or 3390 device. The file access algorithms have been optimized for a cylinder boundary allocation.***

Allocating the CF

You can use any convenient technique to allocate this file. Here is a typical DD statement allocating the Control File:

This JCL is self-explanatory.

☞ ***In a MAS environment, the Control File must not be allocated in a volume that contains JES2 PARMLIB libraries or the JES2 Duplex dataset.***

No other allocation parameters are required.

```
//CF       DD   DSN=cf.filename,DISP=(,CATLG),
//              SPACE=(CYL,64,,CONTIG),UNIT=unitname
```

CF Space Allocation

The file must be cataloged.

The name of the file you chose must be then included in the FILE CF initialization statement.

CF and Performance

The control file must be placed on a device that provides good to excellent service times. Access to the file has been optimized for a cylinder boundary.

The MINDORM, MAXDORM, and MINHOLD keywords are provided to ensure that all systems sharing the file can have access to it. The use of these values is similar to the equivalent parameters in JES2.

ThruPut Manager also supports a MAXHOLD keyword that allows you to specify the maximum time a system can HOLD the Control File. MAXHOLD is intended to control JESplex members that do not run much (or any) batch work. It is a requirement that ThruPut Manager be running on all members of the JESplex but it is not desirable that these non-batch members interfere with Control File activity by any unnecessary accesses.

A recommended set of time values for a quiet or non-batch member is:

```
MINDORM(0) MAXDORM(12000) MAXHOLD(250)
```

This allows the member to be responsive if any activity originates from that member, such as a job arriving or a VARY ONLINE command being issued. Yet it will only access the Control File once every 2 minutes otherwise, and will release it immediately if no work is to be processed.

If your JES2 checkpoint is in OS/390 R4 mode, the DEPTH keyword allows you to control how many JES2 action requests can be queued before ThruPut Manager waits.

These keywords can be specified on the FILE CF initialization statement or on a CFM SET initialization statement within a FOR group. Additionally, these parameters can be changed with the /CFM SET operator command.

For synchronization purposes, the SYNCTOL value of JES2 is used.

CF and Resource Hold

TM/JBS ONLY

If you are running JES2 in OS/390 R4 mode, WLM accounts for jobs held by TM/JBS and TM/JLS in the “operator hold” category, which might not be desirable. You can change this behavior by using the /CFM SET RESOURCEHOLD operator command:

```
/CFM SET RESOURCEHOLD(ON | OFF)
```

RESOURCEHOLD is OFF by default. When it is set to ON, ThruPut Manager prevents TM/JBS and TM/JLS jobs from running by temporarily removing the system affinity. WLM therefore accounts for the delay in the “lack of resource” category.

RESOURCEHOLD can be turned OFF at any time. Affected jobs are returned to the normal ThruPut Manager hold category. If any image of ThruPut Manager must revert to a level without RESOURCEHOLD support, or if any JES2 is UNACTIVATED to pre-R4 mode, you must set RESOURCEHOLD to OFF before reverting to the earlier level.

Formatting the CF

The first time that the Control File is used, TMSS automatically formats the file. TMSS *always* asks for confirmation before formatting the Control File.

Note that this takes place only if the FILE CF initialization statement is present.

☞ ***The contents of this file are critical. Make sure that operating procedures are in place to take care of problems with this file. Only one system should be active at the time this file is formatted.***

A complete description of the operational procedures for formatting the Control File, or if necessary, the individual application areas of the Control File, is contained in the publication ***Operating Guide***.

CF and RESERVE

For performance reasons, do not allow the Control File RESERVE to be converted to a GLOBAL ENQ or propagated as a GLOBAL ENQ. If your installation is using GRS or a similar means of propagating ENQs, do *not* include the CF RESERVE in the RESERVE conversion RNL, but include it in the SYSTEMS exclusion RNL. The qname used by ThruPut Manager for the CF RESERVE is TMRMCF. The rname is formed by the dataset name, padded with blanks to 44 characters, followed by the serial number of the volume containing the Control File.

Sharing the Control File

For each JES2 node, you must share the Control File among all systems that comprise the node.

Do not share the Control File across JES2 nodes. Use multiple Control Files, one per node.

In systems ***with TM/DCS***, use the DCS SET initialization statement to activate XCFM support for cross-systems communications:

```
DCS SET XCF(ON)
```

Note that ThruPut Manager does not allow two active nodes with the same name.

The Control File Utility

The Control File Manager uses area sizes and boundaries optimized either for a 3380 or a 3390 device, depending on the allocation.

☞ ***To move the Control File to another area or another device you must use the Control File Utility.***

The utility DTMCFMUn (where **n** represents the current ThruPut Manager release number) provides the necessary facilities to move the Control File, and also to perform problem analysis and repair.

Installing and Running the Utility

The utility is installed with the Base Product under the name DTMCFMUn. This utility can be used for the following purposes:

1. To move the file to another area on disk, or to a different device type.
2. To produce a formatted dump.

For the move function, you should allocate space for the new Control File. You can use any standard allocation technique. 64 cylinders are needed for a 3380 or 3390, regardless of the device used. The allocation should be on a cylinder boundary and on a device that provides good to excellent service times.

To run DTMCFMUn, you need one of the following sets of JCL. Shown on the next page are examples for copying the file, dumping the file, and for performing both tasks simultaneously.

```
//Name     EXEC PGM=DTMCFMUn,PARM=COPY
//SYSPRINT DD   SYSOUT=*
//DTMRMCF  DD   DSN=name.of.the.control.file,DISP=OLD
//NEWRMCF  DD   DSN=name.of.the.new.file,DISP=(,CATLG),
//              SPACE=(CYL,64,,CONTIG),UNIT=unitname
```

JCL to Copy the Control File

```
//Name     EXEC PGM=DTMCFMUn,PARM=DUMP
//SYSPRINT DD   SYSOUT=*
//DTMRMCF  DD   DSN=name.of.the.control.file,DISP=OLD
//DTMDUMP  DD   SYSOUT=*
```

JCL to Dump the Control File

JCL to Copy and Dump the Control File

Analyzing and Repairing JBS and JLS Control Areas

If you suspect problems with the JBS or JLS control areas, or if you have encountered message DTM6603E, you can use DTMCFMUn to analyze and perhaps repair the damage. To do this, use one of the following PARMs on the EXEC statement:

ANALYZE

This parameter requests an analysis of the JBS and JLS areas of the Control File, and produces a report that is directed to the dataset described by the DD statement DTMRMUP.

`REPAIR`

This parameter performs the same functions as ANALYZE, and also writes a repaired copy of the Control File to a pre-allocated dataset described by the DD statement NEWRMCF.

Note that if the REPAIR function terminates successfully, the resulting Control File will not contain errors, but information about some jobs might be lost. Examine the analysis report to determine which jobs might be affected.

Some examples of the JCL needed to perform analysis and repair are shown below.

The Volume Information File (VIF)

The VIF is a ***mandatory*** shared DASD file which tracks volumes that are being managed by JSS. The VIF is used to mantain information about volumes, including virtual volumes, and their status.

```
//Name     EXEC PGM=DTMCFMUn,PARM=ANALYZE
//SYSPRINT DD   SYSOUT=*
//DTMRMCF  DD   DSN=name.of.the.control.file,DISP=SHR
//DTMRMUP  DD   SYSOUT=*
```

JCL to Analyze the Control File

```
//Name     EXEC PGM=DTMCFMUn,PARM=REPAIR
//SYSPRINT DD   SYSOUT=*
//DTMRMCF  DD   DSN=name.of.the.control.file,DISP=SHR
//NEWRMCF  DD   DSN=name.of.the.new.file,DISP=OLD
//DTMRMUP  DD   SYSOUT=*
```

JCL to Repair the Control File

Implementation Summary

The steps for implementing VIF support are:

- Determine the approximate number of volumes that your installation will be tracking through the VIF. A formula is provided under the heading "Calculating VIF Capacity".
- Calculate the size of the file required. If you plan to write exits that use the user data area, include the size of the area (VSEG) in each record.
- Allocate the file.
- Add the FILE VIF statement to the TMSS initialization statements. If your installation wants to use the user data area, include the parameter. This field is accessible through installation exits only.
- Start TMSS using the COLD parameter for the VIF.
- Confirm the TMSS formatting request by replying 'Y'.

TMSS Initialization Statements

The VIF is a required file. It must be specified with the FILE VIF TMSS initialization statement. The size of the VIF depends on the number of volumes tracked, and the size of the user segment of the record. These parameters are specified using the TMSS FILE VIF statement documented in "Chapter 2. TMSS Initialization Statements ."

VIF Size and Structure

The VIF file consists of a control record, followed by a number of volume records. The maximum number of volume records is defined by the ENTRIES parameter of the TMSS initialization statement FILE VIF.

A diagram of the structure of this file is shown below.

In the VIF:

- The control record indicates the number of volume records being used.
- A volume record contains information about one volume serial number:
  - The volume serial number.
  - The job that requested it.
  - The time of the request.
  - The status of the volume.

- Installation data, if any. The size of the installation data area is defined by the VSEG parameter of the FILE VIF statement.

Allocating the VIF

The VIF can be allocated using any convenient technique. Here is a typical DD statement to allocate the file:

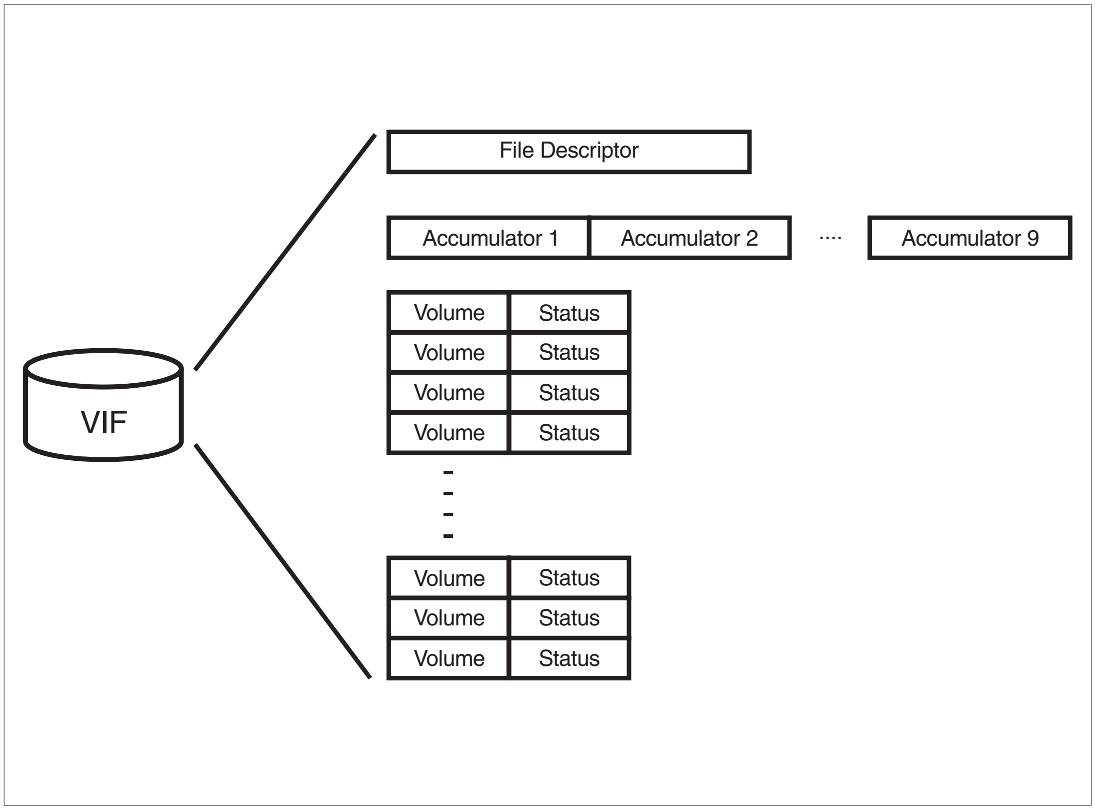
This JCL is self-explanatory.

Note that the number of cylinders required for the VIF depends on the space specified for installation data in the VSEG parameter of the FILE VIF statement. VIF records are 80 bytes long plus any installation data segment.

No other allocation parameters are necessary.

Calculating VIF Capacity

To calculate the space requirements for the VIF, use the following formula:



Volume Information File Structure

$$NumberOfTracks = \left[\frac{NumberOfVolumeEntries \times (80 + VSEG)}{TrackSize} \right] + 1$$

Each individual volume used in a job occupies one *volume entry*, no matter how many references are made to the volume. If the volume is referred to in another job, however, it requires another volume entry. If you are using IBM Virtual Volumes, note that these volumes are added to the VIF.

The first track of the VIF is used to hold the control information, so one is added to the quotient of volumes divided by track capacity.

A single cylinder usually holds a VIF large enough for most installation's needs.

Formatting the VIF

The VIF characteristics are controlled through the TMSS initialization statement described above. The VIF is formatted automatically upon its first use, and any time VIF=COLD is specified when starting TMSS. If the VIF is moved to a different device type, the operator is informed and is given a choice to format the file.

```
//VIF      DD   DSN=vif.filename,DISP=(,CATLG),
//              SPACE=(CYL,n,,CONTIG),UNIT=unitname
```

JCL to Allocate the Volume Information File

TMSS *always* asks for operator conformation before formatting the VIF.

A complete description of the operational procedure for formatting the VIF is contained in the publication ***Operating Guide***.

Moving the VIF

The VIF can be moved among ***identical*** device types. If it is necessary to move the file to a different device type, the VIF must be reallocated and formatted.

VIF and RESERVE

ThruPut Manager issues both a RESERVE and an ENQ for the VIF. For performance reasons, do not allow the VIF RESERVE to be converted to a GLOBAL ENQ or propagated as a GLOBAL ENQ. If your installation is using GRS or a similar means of propagating ENQs, do *not* include the VIF RESERVE in the RESERVE conversion RNL, but include it in the SYSTEMS exclusion RNL.

The ***qname*** used by ThruPut Manager for both the VIF RESERVE and ENQ is DTMVIF. The ***rname*** for the RESERVE is formed by the dataset name, padded with blanks to 44 characters, followed by the serial number of the volume containing the VIF:

```
SYS2.TMV61.VIFILE.PROD     WORK01
```

The ***rname*** for the ENQ is formed in the same way, but LOCK is appended to the volume serial number:

```
SYS2.TMV61.VIFILE.PROD     WORK01LOCK
```

Sharing the VIF

The VIF should be shared across nodes. To allow JSS to hold and release *all* jobs depending on a particular volume, the VIF must be shared by all systems using the same set of volumes. If systems from multiple nodes share the VIF, each node must have a unique name.

SPOOL File

The SPOOL file serves a function for Console Printing Services that is similar to your JES2 Spool. It allows you to place DCS ALERTs in this file so that they can be printed by a single CPS WRITER, thus you only need one CPS printer for your MAS Complex. This file has been designed to be shared.

Spooling gives you flexibility to manage DCS ALERTs. ALERTs are spooled to the SPOOL File, from which they can be selected by a CPS Writer on any system sharing the file, providing that the CPS Writer's selection criteria are met.

Implementation Summary

The implementation steps for the SPOOL file are simple:

- Determine the size of the SPOOL file. Refer to the discussion below.
- Allocate the file.
- Include the FILE SPOOL initialization statement in all systems.
- The first time it is used, code the TMSS start SPOOL=COLD parameter.
- Confirm the TMSS formatting request by replying ‘Y’.

TMSS Initialization Statements

The SPOOL file is defined to TMSS with the FILE SPOOL initialization statement. The statement is described in “Chapter 2. TMSS Initialization Statements.”

SPOOL File Size

The nature of the SPOOL File does not permit an exact calculation of its size. The following points may help in your space allocation decision:

- The minimum space occupied by one record is 1K, and most DCS ALERTs do not exceed this.

Allocating the SPOOL File

The SPOOL File can be allocated using any convenient technique. Typical JCL to allocate the file would include a DD statement such as:

This JCL is self-explanatory.

```
//SPOOL    DD   DSN=spool.filename,DISP=(,CATLG),
//              SPACE=(CYL,n,,CONTIG),UNIT=unitname
```

JCL to Allocate the SPOOL File

No other allocation parameters are necessary.

SPOOL File Limits

ThruPut Manager Spooling supports the following maximums:

- 32 systems
- 64 CPS Writers
- 128 Destinations
- 2048 slots for records

Formatting the SPOOL File

A newly defined SPOOL File is automatically formatted the first time it is used.

The dataset is also formatted if TMSS is started using the SPOOL=COLD parameter. The first TMSS function to open the file triggers the formatting.

TMSS *always* asks for operator confirmation before formatting the SPOOL file.

A complete description of the operational procedure for formatting the SPOOL File is contained in the publication ***Operating Guide***.

☞ ***When multiple CPU installations share a SPOOL File, formatting the file without proper synchronization causes errors. The severity of the error depends on the circumstances. Only one system should be active at the time the SPOOL File is being formatted.***

SPOOL File Structure

The SPOOL File is a DASD file, consisting of a SPOOL Control Record and a record for each DCS Alert waiting to be printed. Once Alert is printed, its SPOOL record is purged. The dataset is formatted with fixed length records.

SPOOL File Considerations

The following points about the SPOOL File should be noted:

- It is confined to a single volume.
- The primary allocation must contain sufficient space, since secondary allocation is not supported.
- All DASD devices supported by MVS are eligible for the SPOOL File *except* the 3340.
- The SPOOL File is completely portable among supported devices. It can be copied to a different device and used by ThruPut Manager without being reformatted.
- The qname used by ThruPut Manager for the SPOOL File RESERVE is DTMJBMQ.

SPOOL File and Reserve

For performance reasons, do not allow the SPOOL RESERVE to be converted to a GLOBAL ENQ or propagated as a GLOBAL ENQ. If your installation uses GRS or a similar means of propagating ENQs, do *not* include the SPOOL RESERVE in the RESERVE conversion RNL, but include it in the SYSTEMS exclusion RNL. The qname used by ThruPut Manager for the SPOOL RESERVE is DTMJBMQ. The rname is formed by the dataset name, padded with blanks to 44 characters, followed by the serial number of the volume containing the SPOOL. file.

Sharing the SPOOL File

The SPOOL File can be shared across nodes. SPOOL File sharing should duplicate VIF sharing; that is, the SPOOL File should be shared by all systems using the same set of volumes. If systems from multiple nodes share the SPOOL File, each node must have a unique name.

The CMF File

This ***optional*** file is used exclusively by DCS to record contention records that are used by the management reports produced by DCS. ***This file is shared by all the systems that are part of the DCS complex.***

Implementation Summary

The implementation steps for the CMF file are simple:

- Determine the size of the CMF file. Refer to the discussion on file size below.
- Allocate the file.
- Include the FILE CMF initialization statement.
- The first time the CMF file is used, DCS detects that the file has not been formatted. The formatting process is automatically initiated.
- Confirm the TMSS formatting request by replying 'Y'.

TMSS Initialization Statements

The CMF file is defined to TMSS with the FILE CMF initialization statement. The statement is described in "Chapter 2. TMSS Initialization Statements."

CMF File Size

The nature of the CMF File does not permit an exact calculation of its size. The following points may help in your space allocation decision:

- This file is a "wrap-around" file, that is, you do not run out of space. You can, however, lose the oldest data if the file wraps before you have collected the data for your reporting system.
- A "snap-shot" utility is provided to transfer the records from this shared file to another external file suitable for input to the management reporting system.

- One cylinder of a 3380/3390 device holds records for approximately 600 to 1000 jobs in contention. The precise number cannot be established because the number of datasets and TSO holders varies from situation to situation.
- This data is not critical so the consequences of losing some records should not be severe.
- Establish a cycle for data “unloading” (daily, weekly, ...) that suits your installation. Calculate the number of jobs that might run into contention during that cycle time, then double the space needed. This approach should provide you with a comfortable margin of error.

Allocating the CMF File

The CMF File can be allocated using any convenient technique. Typical JCL to allocate the file would include a DD statement such as:

```
//TMCMF    DD   DSN=cmf.filename,DISP=(,CATLG),
//              SPACE=(CYL,n,,CONTIG),UNIT=unitname
```

JCL to Allocate the CMF File

The JCL shown above is self-explanatory.

Formatting the CMF File

The dataset is formatted if DCS detects that it has not been formatted.

TMSS asks for operator confirmation before formatting the CMF file.

A complete description of the operational procedure for formatting the CMF File is contained in the publication ***Operating Guide***.

CMF File Structure

The CMF File is a DASD file, consisting of an CMF Control Record and a variable number of dataset contention records. The control record points to where the next record is to be written.

The structure of the file prevents the situation where you run out of space by using a “wrap-around” technique.

CMF File Considerations

The following points about the CMF File should be noted:

- It is confined to a single volume.
- The primary allocation must contain sufficient space, since secondary allocation is not supported.
- All DASD devices supported by MVS are eligible for the CMF File *except* the 3340.
- DCS formats it as follows:

```
DSORG   PS
RECFM   F
LRECL   4096
```

- The CMF File is completely portable among supported devices. It can be copied to a different device and used by ThruPut Manager without being reformatted.

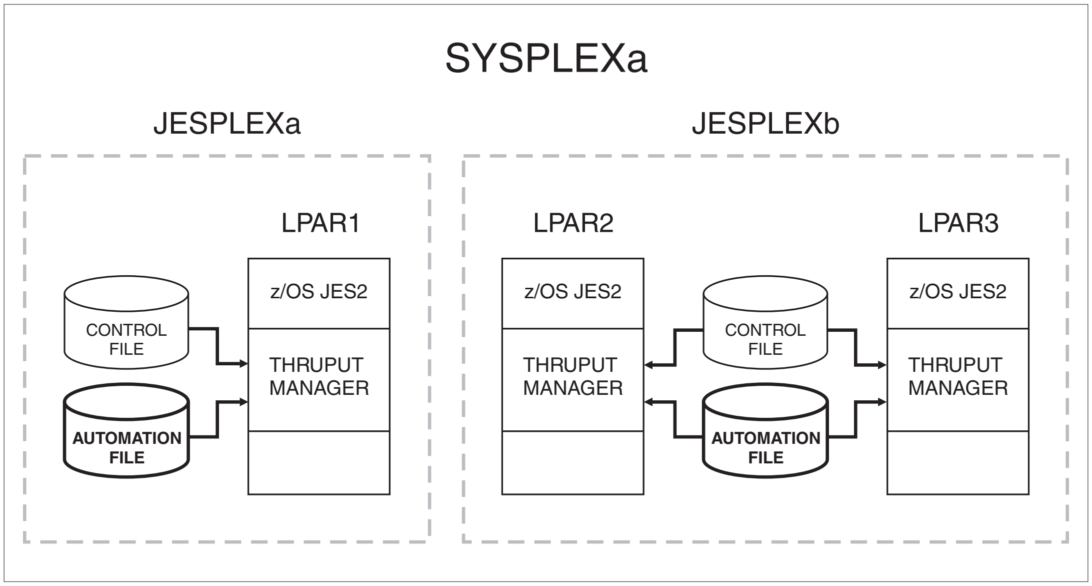
CMF File Sharing

CMF File sharing should duplicate Control File sharing; that is, the CMF File should be shared to include all systems in a particular JES2 node, and there should be one CMF file per JESplex.

The Automation File

TM/DBS ONLY

The ***Automation File*** is required by TM/DBS. It is shared among all systems in a JESplex. Installations with more than one JESplex will need ***one Automation File for each JESplex.***



Each JESplex requires an Automation File

Automation Edition components use the Automation File to store Configurations and Policies. The Automation File is initialized and maintained using ISPF dialogues. When you define an Automation File for the first time, information describing the JESplex is collected. Since this description is used by other automation facilities, the Automation File for a particular JESplex needs to be created and initialized only once.

There are no performance considerations. You need only specify a volume that is shared across the JESplex. The ISPF dialogue performs the actual allocation and initialization. For details, see the ***Drive Booking Services: System Programming Guide***.

Facilities Summary

FILE Initialization Statements		
Statement	**Description**	**Page**
CFM SET	Sets performance values for the Control File within a FOR group.	38
DCS SET	Controls XCFM support for TM/DCS.	45
FILE CF	Defines the Control File.	50
FILE CMF	Defines the CMF file.	52
FILE SPOOL	Defines the SPOOL File.	53
FILE VIF	Defines the Volume Information File.	54

CFM Operator Commands	
Command	**Description**
CFM DELETE NODE	Removes a JES2 node from the Control File.
CFM DISPLAY	Displays information about the Control File.
CFM SET	Sets performance values for the Control File.

Chapter 4. Printing Services

This chapter describes the two different types of printing services provided with ThruPut Manager and the different considerations associated with each one.

Introduction

Your installation, through JAL, may instruct the following ThruPut Manager components to generate “small reports”:

- You can direct DCS to generate ALERTs whenever a particular type of work cannot access a dataset after an installation defined time.

For the purpose of discussing Printing Services and ALERTs will be referred to as ***listings*** or ***output***.

ThruPut Manager provides two different mechanisms to print these listings:

1. Console Printing Services (CPS) uses writers and the ThruPut Manager SPOOL file to print on ***hard-copy console devices***.
2. SYSOUT Printing Services (SPS) uses standard JES2 SYSOUT facilities to print on ***JES2 managed printers***.

We do not recommend one method over the other. From a definition and management point of view, the SPS mechanism is simpler because it uses standard JES2 facilities. However, if you do not “isolate” your SYSOUT listings they may get lost among other JES2 listings because they are small.

The two different methods are described below.

Console Printing Services (CPS)

CPS provides printing services using hard-copy console devices. There are two facilities to support CPS:

- The CPS SPOOL File.
- The CPS Writer.

They are explained below.

The SPOOL File

The CPS SPOOL File, as its name indicates, is used to spool ThruPut Manager generated listings. In this way, they do not have to be printed directly. Spooling provides several advantages; however, its most obvious use is in multisystem environments. Without the spooling facility, one CPS hard-copy console device would be required per system. We strongly recommend that you do not attempt to use CPS without using the SPOOL File.

The structure of the SPOOL is shown below:

CPS Writers

After a listing has been created and spooled you want it printed. CPS Writers provide that function.

CPS Writers are defined using TMSS initialization statements. They can be automatically started at TMSS startup time or any time after with operator commands. You can have more than one CPS Writer.

The relationship between a listing, a Writer, and the physical hard-copy console where the listing is actually printed is as follows:

- When a listing is requested in JAL, you assign a DESTINATION. It can be from 1 to 8 characters and represents a “name” chosen by your installation.
- All the listings that are assigned a particular destination are placed in the same queue in chronological order.
- When a CPS Writer is started, it is given one or more destinations from which to select listings for printing.
- The CPS Writer is also assigned a physical console, normally a hard-copy device. To assign the device you can use its UCMID, a route code, or a console name.

CPS Implementation Summary

Before you can take any specific action to implement the facilities offered by CPS you must determine the kind of printing and system sharing requirements for the listings. Your implementation process may take these steps:

- Determine the number of CPS Writers needed. In most situations only one is needed. ThruPut Manager supports up to 64 concurrent CPS Writers.

- If multiple Writers are needed, determine the characteristics of each Writer. For each CPS Writer, add a CPS WRITER statement to the TMSS initialization statements:

  ☞ ***Each CPS Writer using the same SPOOL File must have a unique name:***
  1) If a duplicate Writer name is specified in the same system, TMSS issues an error message and terminates.
  2) If the duplication is detected across systems TMSS issues a warning message, suppresses the duplicate Writer, and continues processing.

- Activate the Spooling mechanism:
  - Allocate the SPOOL File as documented in "Chapter 3. File Definition Services (FDS) Function." One SPOOL File serves all the CPS Writers.
  - Add the FILE SPOOL statement to the TMSS initialization statements.
  - The first time a newly defined CPS SPOOL File is used, the file is automatically formatted (after operator confirmation) ***by the first system*** that starts TMSS.

CPS Dependencies

Hardware

To print using CPS you need a hard-copy device such as an IBM 3287, or its equivalent. The console could be shared with other tasks if necessary, but this is not recommended.

SPOOL File

If CPS Spooling is desired the SPOOL File must be available. For a description of this file refer to "Chapter 3. File Definition Services (FDS) Function."

Page Length Considerations

The printing format is controlled by the CPS WRITER keywords PAGELENGTH, TOPMARGIN, and BOTTOMMARGIN. The values for these keywords are expressed in units of lines. For example, for an 11 inch page printing 6 lines per inch, PAGELENGTH is 66.

The combination of keywords required is determined by the type of forms on which the output is printed:

- Continuous roll (no perforations).
- Fanfold sheets or a roll with perforations.

Normally you use continuous roll mode when you want a variable length listing to tear from the printer. In that case, you want to leave space at the top and the bottom so no line of text is partially “cut”. Use the TOPMARGIN and BOTTOMMARGIN keywords in this case. You do not have to specify the PAGELENGTH keyword.

For fanfold paper or a roll with perforations, you define PAGELENGTH to be the total number of lines that can be printed in a page. In addition to PAGELENGTH, TOPMARGIN is all that is required. CPS prints the number of lines indicated with the keyword TOPMARGIN before starting a new listing, and then uses PAGELENGTH to determine the number of blank lines needed to get to the top of the next available sheet.

The number of text lines that are printed on each page is:

```
lines of text = PAGELENGTH - (TOPMARGIN + BOTTOMMARGIN)
```

Spacing to a new page is automatic as soon as the number of lines specified in PAGELENGTH is reached.

SYSOUT Printing Services (SPS)

SPS provides the services needed to use JES2 SYSOUT to print Volume Lists and ALERTs. Since Listings produced using SPS are simply JES2 SYSOUT datasets, you can manage them accordingly.

SPS Implementation Considerations

The characteristics of the SYSOUT output are defined using the SPS DEFINE initialization statement. For a detailed description of the syntax of this statement, refer to “Chapter 2. TMSS Initialization Statements.”

SPS DEFINE statements assign a name to a set of JES2 SYSOUT characteristics. The link between this name and the JAL generated listing is the ***destination name***. The characteristics that can be specified include:

- SYSOUT class.
- The forms to be used.
- Whether the output is to be held.
- The JES2 destination.
- The top, bottom, and left margins.
- The page length.

Because JES2 facilities are used to handle spooling and printing, the implementation considerations for SPS are simple:

- Sharing is dictated by the way in which JES2 MAS is configured.
- Printing is a function of the SYSOUT characteristics assigned to each destination name. Listings, via the Destination, can be directed to different SYSOUT classes, different JES2 destinations, request different forms, or any combination that suits your needs.

For each unique set of requirements, add a corresponding SPS DEFINE statement to the TMSS initialization statements.

SPS Operational Considerations

The SYSOUT characteristics defined with an SPS DEFINE statement can be changed or completely removed through operator commands:

- SPS REMOVE removes the SYSOUT definition completely.
- SPS DEFINE can add a new definition. If a SYSOUT definition has unwanted individual characteristics, you can remove its definition, then define it as desired.

The relationship between a listing and the physical printer where it is actually printed is as follows:

- The DESTINATION assigned to a listing ***is its link*** to a set of SYSOUT characteristics defined by the 'SPS DEFINE *name*' initialization statement or by an 'SPS DEFINE *name*' operator command.
- All the listings using the same ThruPut Manager destination name are processed in chronological order.
- With the SPS DEFINE you assign a ***JES2 class and a JES2 destination*** to the JAL generated listings.
- These listings are selected and printed in the same way any other SYSOUT file is handled. ***Any printer associated with the JES2 destination and the class assigned by the SPS definition*** can print the output.

Other operational considerations for managing the printing output produced by SPS are identical to those for any JES2 SYSOUT datasets.

CPS Versus SPS

- Whether a JAL generated listing is handled by CPS or SPS is determined by the destination associated with the listing. Both methods can co-exist. However, if you intend to use both methods then ***you should understand how the destination "link" works*** in a mixed environment:

- ALERTs have a destination assigned in JAL.
- Printing Services, ***based on the destination***, has to decide to which service to direct the ALERT: is it CPS or SPS?:
    - If there is an SPS definition statement that matches the destination name, then clearly it belongs to SPS.
    - If no matching definition name is found then:
        - If the SPOOL File is available, it is placed in the CPS SPOOL. If no CPS WRITER is active with that destination, the Volume List or ALERT simply stays in the SPOOL waiting to be printed.
        - If the SPOOL File is not defined, it is passed to SPS. Since no definition is available to provide its JES2 characteristics, *it is assigned the default destination $DEFAULT, which has the characteristics of class A, and placed in hold status*. It is then directed to JES2 like any other SYSOUT file.

TMSS Initialization Statements

For Console Printing Services (CPS)

There are two initialization statements:

- CPS WRITER defines and activates CPS Writers. You must code one statement per Writer.
- FILE SPOOL defines the file used to spool output written using CPS Writers.

These statements are documented in “Chapter 2. TMSS Initialization Statements.”

For SYSOUT Printing Services (SPS)

There is one initialization statement:

- SPS DEFINE specifies the characteristics of the SYSOUT output.

This statement is documented in “Chapter 2. TMSS Initialization Statements.”

Facilities Summary For CPS

CPS Initialization Statements		
Statement	**Description**	**Page**
CPS WRITER	Defines and activates CPS Writers.	40
FILE SPOOL	Defines the SPOOL File for CPS Writers.	53

CPS Operator Commands	
Command	**Description**
CPS DISPLAY	Display the status of Writers, destinations, and other CPS related items.
CPS WRITER	Allows you to control CPS Writers. If CPS Spooling is active, any changes made with this command are checkpointed and thus become permanent changes.

Facilities Summary For SPS

SPS Initialization Statements		
Statement	**Description**	**Page**
SPS DEFINE	Defines a set of characteristics to print Volume Lists and Alerts.	63

SPS Operator Commands	
Command	**Description**
SPS DEFINE	Defines a set of characteristics to print Volume Lists and Alerts.
SPS DISPLAY	Displays the characteristics associated with an SPS destination.
SPS REMOVE	Removes an SPS definition.

Chapter 5. User Display Facility (UDF)

This chapter provides the background needed to implement the User Display Facility.

Introduction

The ThruPut Manager User Display Facility (UDF) allows your datacenter users to display information about jobs processed by ThruPut Manager. Using UDF, you can display information such as hold reasons and job dependencies for individual jobs or lists of jobs. This facility runs under ISPF and can be invoked with a single keystroke by assigning the TMUSER command to a PF key, referred to as the Hot Key.

☞ ***In order to use this facility, the appropriate ISPF datasets must be concatenated during the INSTALL process described in the Installation Guide.***

UDF is sensitive to the environment in which it is invoked, and within recognized environments is also sensitive to the position of the cursor. Supported spool management products include:

- SDSF
- IOF
- JES Master
- CA/SYSVIEW

Once invoked, UDF uses a pop-up window approach to show you information about jobs processed by ThruPut Manager.

UDF can be added to an ISPF environment quickly and easily. UDF uses standard ISPF interfaces and requires no modifications to the system.

Some Concepts

UDF is invoked through an ISPF command: TMUSER. It uses a PC-like interface to give users instant and convenient access to needed information.

In order to understand how UDF resembles a PC-like interface, it is necessary to introduce some basic concepts before proceeding with detailed descriptions.

Windows

A *window* is a framed portion of the logical ISPF screen that displays ThruPut Manager information.

A UDF pop-up window can be thought of as a mini-ISPF screen. For most purposes, therefore, UDF windows can be treated as if they were ISPF screens. When UDF opens a window, it respects the logical ISPF screen divisions you have selected. If you split the screen, UDF detects this and adjusts its windows to respect your boundaries.

UDF operates with the standard ISPF scrolling commands. Standard ISPF scrolling commands and techniques are also supported at the command prompt. You can scroll up and down the display in any window that shows the word MORE in either the upper or lower line of the frame. The amount of text scrolled down or up can be controlled through the SCROLL field at the top of the logical screen.

The Hot Key

The *Hot Key* is a PF key that has been assigned the command TMUSER, which invokes UDF.

Like any command used in ISPF, you can assign TMUSER to a key of your choosing. Doing so allows you to display information by simply placing the cursor and hitting the assigned key. Although it is not necessary to assign a PF key to TMUSER, doing so makes invoking UDF easier and allows you take full advantage of its ability to detect supported spool management products.

The following discussions assume that such a Hot Key has been defined, although in every case you can achieve the same effect by entering the TMUSER command at the ISPF command prompt, then moving the cursor to the desired location before hitting ENTER.

Invoking the User Display Facility

The User Display Facility can be invoked in two ways:

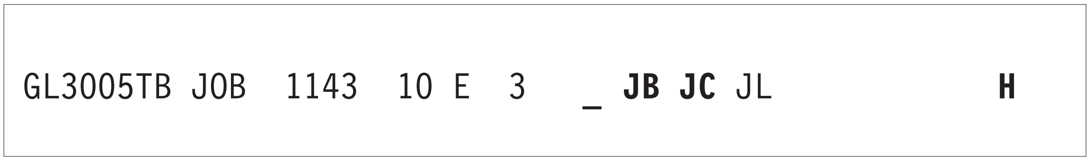
- *In-context*, by hitting the UDF Hot Key or by entering the TMUSER command while using a spool management product supported by UDF.
- *Explicitly*, by entering the TMUSER command anywhere else in ISPF.

These invocation types are explained in detail later, but first it is necessary to introduce the *Information Summary Line* concept.

The Information Summary Line

Regardless of the type of invocation, UDF takes you to a window that has one or more Information Summary Lines. The Information Summary Line for a job ***always displays the same format and can always be used in the same way***, no matter which window displays it.

The Information Summary Line contains the jobname and job number. This might be followed by a line command area, depending upon the job's status.



```
GL3005TB JOB  1143  10 E  3   _ JB JC JL          H
```

Sample Information Summary Line

ThruPut Manager services affecting the job are listed using two letter acronyms. For example JB for Job Binding Services. Highlighted acronyms indicate that the service is causing the job to be held.

Finally, the line may contain the letters **H** or **D**, showing whether the job has been held or deferred by ThruPut Manager.

Using the Information Summary Line

You can think of the Information Summary Line as a control line that you can use to get to any UDF window that applies to the job, as long as you are authorized to access the detailed information.

The ability to display the detailed level of job information can be restricted to specific ISPF users. How to control the display of information is discussed later on in this chapter.

The Line Command Area

If there is ThruPut Manager information available, the Information Summary Line has a line command area:

- Entering a **D** opens the Job Detail Window.
- Entering a **I** opens the JES2 Job Select Information Window.
- Entering a **V** opens the Volume Information Window.

These windows are described in detail later.

The Service Acronyms

On the Information Summary Line, every ThruPut Manager service affecting a job is represented by its two letter acronym. For example, a job using Job Setup Services has the acronym JS displayed. To display further information about a service and how it is affecting a job, tab or move the cursor to the acronym for the service in which you are interested, then hit ENTER. UDF opens the window associated with that service.

Other Information Displayed

The following descriptions explain other information that could be displayed on the Information Summary Line for an individual job.

D

If the job has been deferred, it is flagged with the letter D.

H

If the job is in hold, it is flagged with the letter H. If the job is being held by one of the ThruPut Manager services, the acronym for that service is highlighted.

Data Only

This job is not under the control of any ThruPut Manager Services, therefore only general information or Volume information (if any) can be displayed.

Failed

This job was failed during job analysis. ThruPut Manager information could be available.

Exempt

This job was submitted in a class that is exempt from ThruPut Manager processing, therefore no ThruPut Manager information is available.

Analyzing

This job is currently being analyzed by ThruPut Manager. If the job has not been previously analyzed, no ThruPut Manager information is available.

Awaiting Analysis

This job is awaiting analysis by ThruPut Manager. If the job has never been analyzed, no ThruPut Manager information is available.

Awaiting XMIT

This job is awaiting transmission to another node. No ThruPut Manager information is available.

Information Not Yet Available

There is no ThruPut Manager information currently available for this job. The job has been selected while on the input or converter queue. Line commands are accepted because the job's status will change.

Not ThruPut Manager Processed

ThruPut Manager has never processed this job, therefore no information is available.

Duplicate

You have scrolled the spool manager's screen left or right, therefore UDF can no longer distinguish between duplicate jobnames. ThruPut Manager information cannot be provided until you provide a job number.

Not Found

The job has been purged since the last time the spool manager's display was refreshed. No ThruPut Manager information is available.

Job Analyzer Bypassed

The job has bypassed the job analyzer as a result of operator intervention. No ThruPut Manager information is available.

In-context Invocation

UDF supports invocation *in-context* when used in conjunction with these spool management products:

- SDSF, a product of IBM.
- IOF, a product of Fischer International Systems Corporation.
- JES Master, a product of XENOS Group Inc.

When invoked in-context, UDF operates in one of two modes:

- List Mode, which provides a summary for each job in a list. The information is shown in the Job List Window.
- Job-specific Mode, which provides specific detailed information for a particular job. The information is shown in a variety of windows, depending on the request.

Explicit Invocation

You can invoke UDF *explicitly* from anywhere in ISPF by hitting the Hot Key or entering the TMUSER command. The first window opened by explicit invocation is the Job Prompt Window, and as its name indicates it prompts you to enter the job name and job number. You must enter enough information to identify a specific job: a unique jobname; a job number; or if there are duplicate jobnames, a jobname and number combination.

Once a specific job has been identified, UDF opens the Job Detail Window *provided that you are allowed to view detailed information for the job*, otherwise a window containing the Information Summary Line and a message is opened.

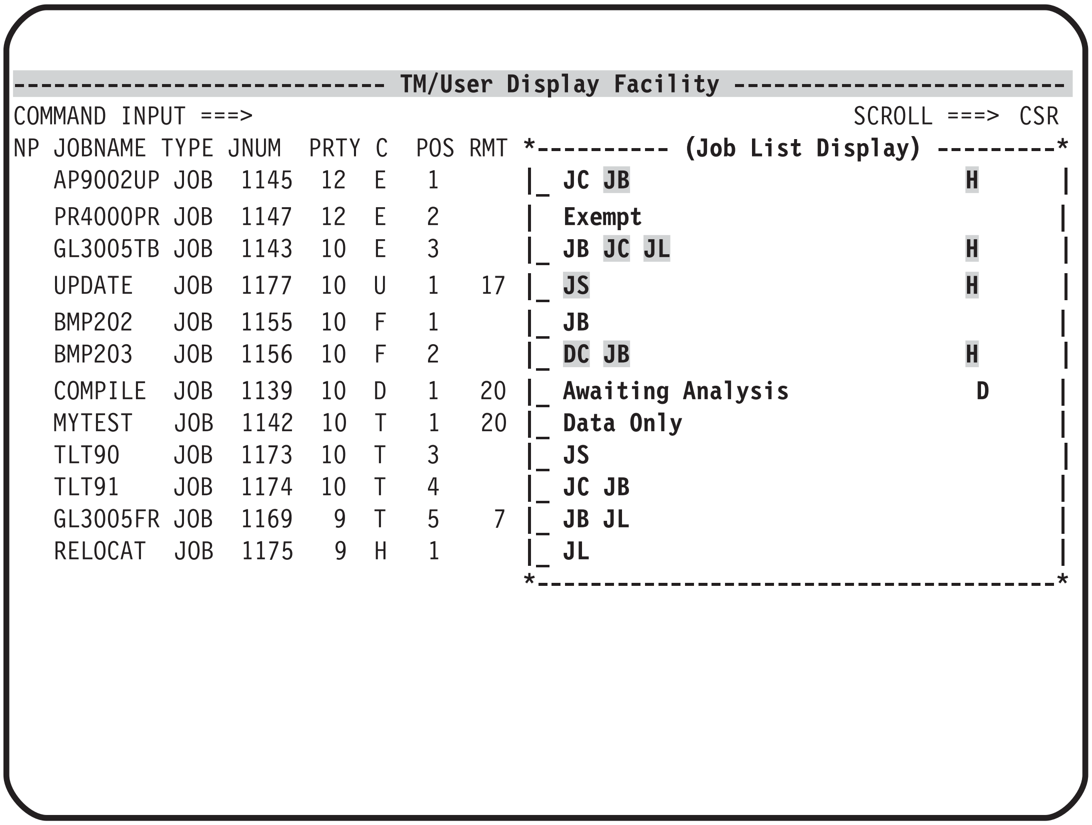
When invoked explicitly, UDF operates in Job-specific Mode.

List Mode

As indicated earlier, *List Mode* occurs when UDF is invoked in-context. In this mode, UDF displays a list of Information Summary Lines corresponding to a list of jobs displayed by a supported spool manager.

When operating in *List Mode* UDF uses a concept called a *coupled window*. The Job List Window, shown below, is directly coupled to the job list display of the spool manager you are using. Each line in the Job List Window is directly related to the same line in the spool manager's display, and if you scroll up or down, the Job List and the spool manager's display scroll together.

The Job List Window



```
---------------------------- TM/User Display Facility -------------------------
COMMAND INPUT ===>                                                  SCROLL ===>  CSR
NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                        H       |
   PR4000PR JOB  1147  12  E   2      |  Exempt                               |
   GL3005TB JOB  1143  10  E   3      |_ JB JC JL                     H       |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                           H       |
   BMP202   JOB  1155  10  F   1      |_ JB                                   |
   BMP203   JOB  1156  10  F   2      |_ DC JB                        H       |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis             D      |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                            |
   TLT90    JOB  1173  10  T   3      |_ JS                                   |
   TLT91    JOB  1174  10  T   4      |_ JC JB                                |
   GL3005FR JOB  1169   9  T   5    7 |_ JB JL                                |
   RELOCAT  JOB  1175   9  H   1      |_ JL                                   |
                                      *---------------------------------------*
```

The Job List Window

How Is This Window Opened?

The Job List Window opens when you hit the Hot Key while you are displaying a job list from one of the spool managers *but **without** placing the cursor on a line containing a specific jobname or job number*. Whenever the Job List Window is active, you are in List Mode.

What Is In This Window?

The Job List Window simply contains a corresponding Information Summary Line for each job listed in the spool manager's display.

From This Window: Where Can I Go?

From the Job List Window you can go:

- To the Job Detail Window by entering **D** in the line command area of an Information Summary Line and hitting ENTER.
- To the Volume Information Window by entering **V** in the line command area of an Information Summary Line and hitting ENTER.
- To the JES2 Job Select Information Window by entering **I** in the line command area of an Information Summary Line and hitting ENTER.
- You can go to the window for any ThruPut Manager service *for which an acronym is shown* by tabbing to the particular acronym in an Information Summary Line and hitting ENTER.

Job-specific Mode

Job-specific Mode is active whenever you are using a UDF window to display details about an individual job. Various UDF windows display information about different aspects of a specific job and how ThruPut manager processing affects it:

- The *Job Detail Window* displays details about a job have been associated with a job as a result of ThruPut Manager processing. The selection of job details is under the control of your installation.
- The *Volume Information Window* displays a summary of the volumes associated with the job at the time of ThruPut Manager processing.
- *TM Services Display Windows* display details about a specific ThruPut Manager service and how it affects the job. The ThruPut Manager base product includes:
  - The Job Setup Services Display Window.

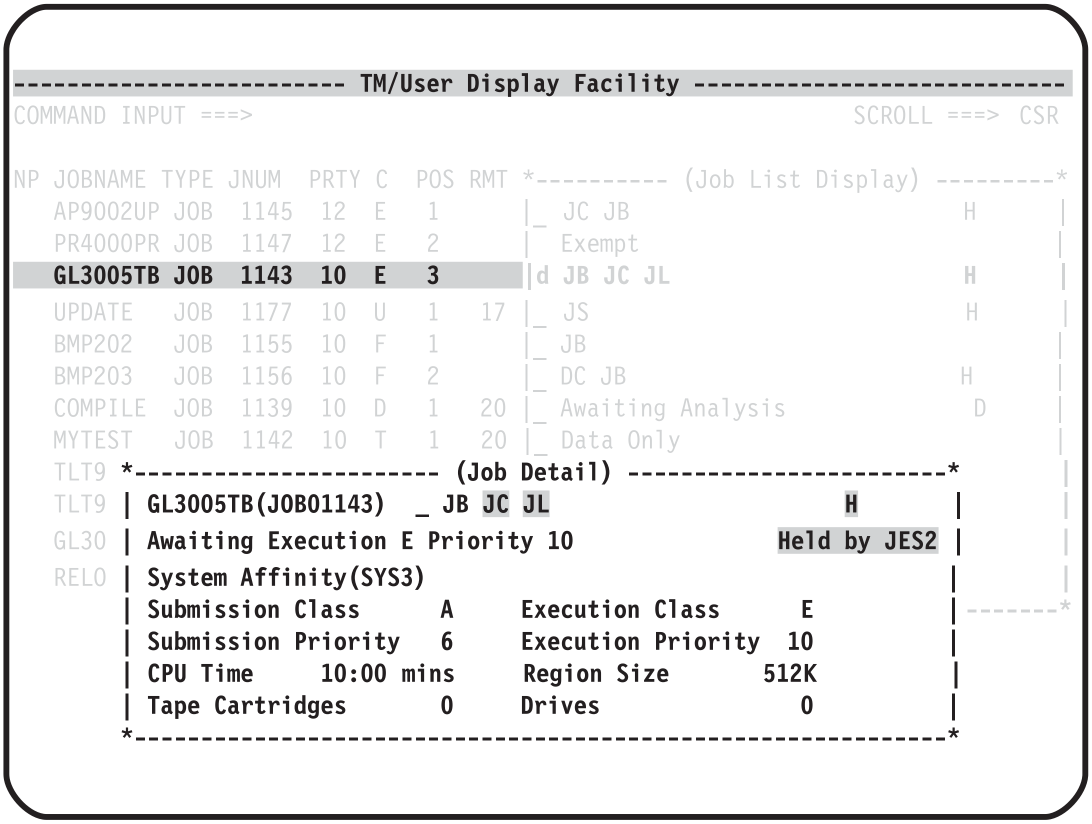
  When optional ThruPut Manager components are installed, additional windows become available:

  - The Dataset Contention Services Display Window.
  - The Job Binding Services Display Window.
  - The Job Chaining Services Display Window.
  - The Job Limiting Services Display Window.
  - The System Level Manager Display Window.

To display information about the ThruPut Manager Services that are currently controlling the job, tab or move the cursor to the acronym in the Information Summary Line for the service in which you are interested, then hit ENTER. UDF opens the window associated with the particular service.

The following pages provide more information about the UDF detail Windows. Note that these windows are available regardless of the type of invocation (explicit or in-context). For details about the windows for the optional ThruPut Manager components, see the chapters describing the individual functions.

The Job Detail Window



```
------------------------- TM/User Display Facility ----------------------------
COMMAND INPUT ===>                                              SCROLL ===>  CSR

NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                       H      |
   PR4000PR JOB  1147  12  E   2      |  Exempt                             |
   GL3005TB JOB  1143  10  E   3      |d JB JC JL                    H      |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                          H      |
   BMP202   JOB  1155  10  F   1      |_ JB                                 |
   BMP203   JOB  1156  10  F   2      |_ DC JB                       H      |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis            D     |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                          |
   TLT9 *---------------------- (Job Detail) ------------------------*      |
   TLT9 | GL3005TB(JOB01143)  _ JB JC JL                      H       |      |
   GL30 | Awaiting Execution E Priority 10              Held by JES2 |      |
   RELO | System Affinity(SYS3)                                      |      |
        | Submission Class      A     Execution Class      E         |-------*
        | Submission Priority   6     Execution Priority  10         |
        | CPU Time     10:00 mins     Region Size      512K          |
        | Tape Cartridges       0     Drives               0         |
        *------------------------------------------------------------*
```

The Job Detail Window

How Is This Window Opened?

To open the Job Detail Window, you do one of the following:

From the UDF Job List Window:

- Tab or move the cursor to the line command area of the Information Summary Line for a job, then enter a **D** and hit ENTER.

Directly from your spool manager window:

- You place the cursor on a line from the spool manager's job list that contains a specific jobname or job number, then hit the Hot Key.
- You do one of the following:
  - For SDSF, you hit the Hot Key when you are on the *Dataset Display Screen*.
  - For IOF, you hit the Hot Key when you are on the *Job Summary Screen*.
  - For JES Master, you hit the Hot Key when you are on the *Job Display* screen.

From the Job Prompt Window:

- Once a specific job has been identified, UDF opens the Job Detail Window provided that you are allowed to view detailed information for the job.

If you are not allowed to view detailed information about the job, a window is opened displaying the Information Summary Line for the job, accompanied by a message.

What Is In This Window?

The first line of the Job Detail Window is the Information Summary Line for the job.

The second line shows the job's current status. If the job is awaiting execution, the third line displays the job's system affinity.

Subsequent lines display information that has been associated with the job as a result of ThruPut Manager processing. The information passed to UDF combines text with information about the job that has been collected by the ThruPut Manager. Your installation can customize this display. If customization has not been done, UDF displays information that is of general interest.

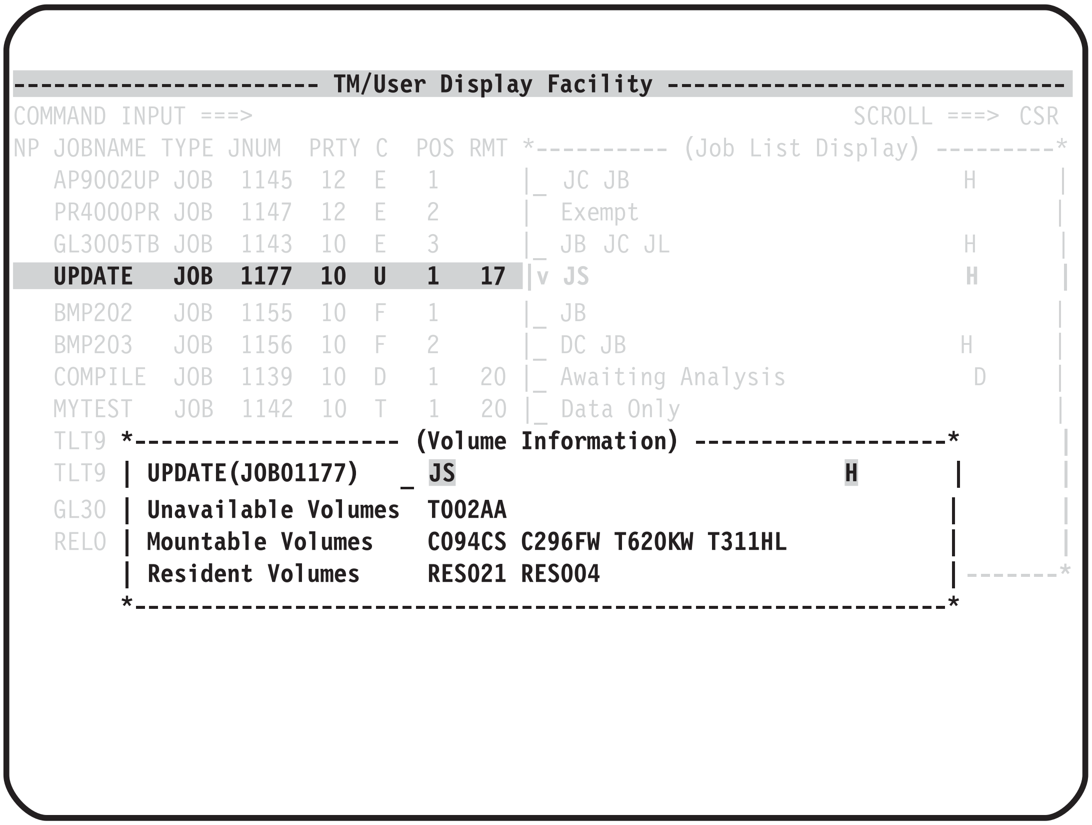
From This Window: Where Can I Go?

From the Job Detail Window you can go:

- To the Volume Information Window by entering **V** in the line command area of the Information Summary Line and hitting ENTER.

- To the JES2 Job Select Information Window by entering **I** in the line command area of the Information Summary Line and hitting ENTER.
- To the window for any ThruPut Manager service for which an acronym is shown by tabbing to the particular acronym in the Information Summary Line and hitting ENTER.
- *If you invoked the Job Detail Window from the Job List Window*, you can return to the Job List Window by hitting PF3, the END key.

The Volume Information Window



```
----------------------- TM/User Display Facility -----------------------------
COMMAND INPUT ===>                                              SCROLL ===>  CSR
NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                      H       |
   PR4000PR JOB  1147  12  E   2      |  Exempt                             |
   GL3005TB JOB  1143  10  E   3      |_ JB JC JL                   H       |
   UPDATE   JOB  1177  10  U   1   17 |v JS                         H       |
   BMP202   JOB  1155  10  F   1      |_ JB                                 |
   BMP203   JOB  1156  10  F   2      |_ DC JB                      H       |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis           D      |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                          |
   TLT9 *-------------------- (Volume Information) -------------------*     |
   TLT9 | UPDATE(JOB01177)   _ JS                              H       |     |
   GL30 | Unavailable Volumes  T002AA                                 |     |
   RELO | Mountable Volumes    C094CS C296FW T620KW T311HL            |     |
        | Resident Volumes     RES021 RES004                          |-------*
        *-------------------------------------------------------------*
```

The Volume Information Window

How Is This Window Opened?

To open the Volume Information Window, type a **V** in the line command area of the Information Summary Line for the job and then press ENTER.

What Is In This Window?

The first line of the Volume Information Window is the Information Summary Line for the job.

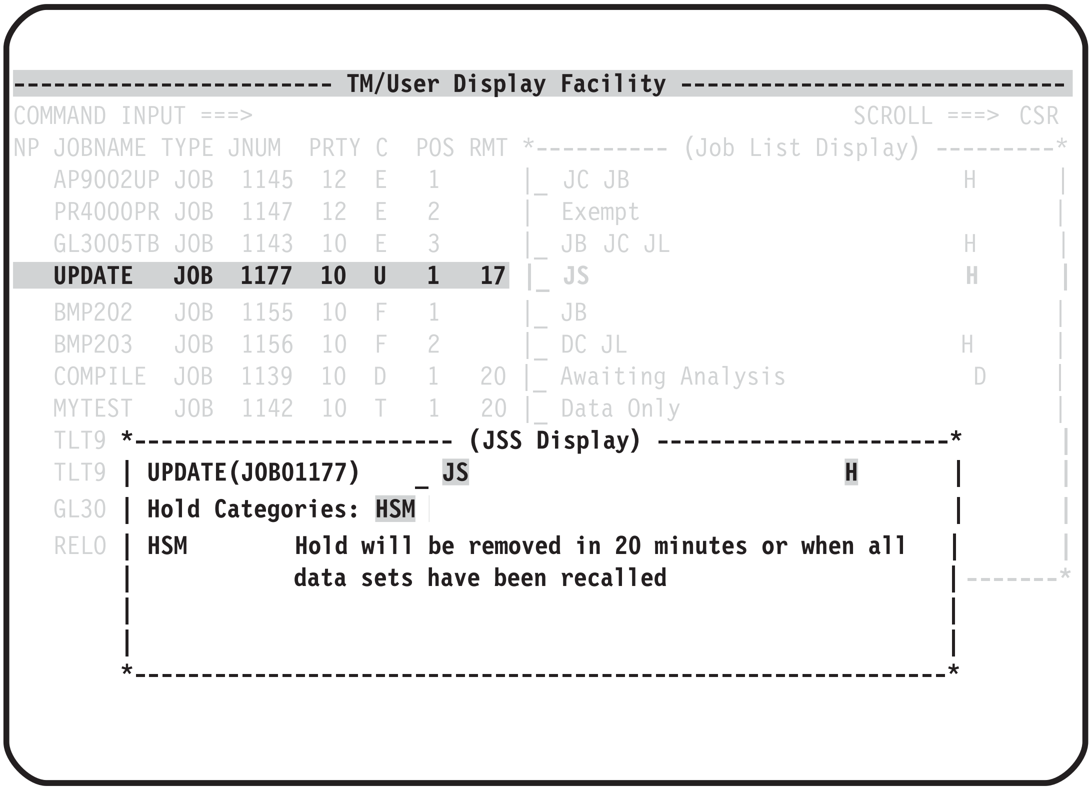
Subsequent lines list the volumes associated with the job at the time the ThruPut Manager processed it.

From This Window: Where Can I Go?

From the Volume Information Window you can go:

- To the Job Detail Window by entering **D** in the line command area of an Information Summary Line and hitting ENTER.
- To the window for any ThruPut Manager service for which an acronym is shown by tabbing to the particular acronym in the Information Summary Line and hitting ENTER.
- *If you invoked the Volume Information Window from the Job List Window*, you can return to the Job List Window by hitting PF3, the END key.

Job Setup Services Display Window



```
------------------------ TM/User Display Facility ----------------------------
COMMAND INPUT ===>                                              SCROLL ===>  CSR
NP JOBNAME TYPE JNUM  PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                      H      |
   PR4000PR JOB  1147  12  E   2      |  Exempt                            |
   GL3005TB JOB  1143  10  E   3      |_ JB JC JL                   H      |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                         H      |
   BMP202   JOB  1155  10  F   1      |_ JB                                |
   BMP203   JOB  1156  10  F   2      |_ DC JL                      H      |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis           D     |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                         |
   TLT9 *------------------------ (JSS Display) ----------------------*    |
   TLT9 | UPDATE(JOB01177)    _ JS                           H       |    |
   GL30 | Hold Categories: HSM                                     |    |
   RELO | HSM        Hold will be removed in 20 minutes or when all  |    |
        |            data sets have been recalled                  |-------*
        |                                                          |
        |                                                          |
        *----------------------------------------------------------*
```

The JSS Display Window

How Is This Window Opened?

The JSS Window is opened from the Information Summary Line in any window by tabbing or moving the cursor to the JSS acronym and hitting ENTER.

What Is In This Window?

The first line of the JSS Window is the Information Summary Line for the job.

The next line lists the hold categories for the selected job.

For each hold category listed, a subsequent text line provides further explanation of the hold. Remember that jobs can also be held for reasons not summarized in this window.

From This Window: Where Can I Go?

From the JSS Window you can go:

- To the Job Detail Window by entering **D** in the line command area of the Information Summary Line and hitting ENTER.
- To the Volume Information Window by entering **V** in the line command area of the Information Summary Line and hitting ENTER.
- To the window for any ThruPut Manager service for which an acronym is shown by tabbing to the particular acronym in the Information Summary Line and hitting ENTER.
- *If you invoked the JSS Window from the Job List Window*, you can return to the Job List Window by hitting PF3, the END key.

Managing the User Display Facility

The User Display Facility allows users to examine information that has been collected by ThruPut Manager. You can also store information with each job to be displayed by users.

New Job Action Language statements are available with UDF, providing these functions:

- A way to store information with each job. Authorized users can then use the TMUSER command to display the accumulated data.
- A means to control which users are authorized to display information.

The ability to display the detailed level of information for a job can be restricted to specific ISPF users. Through the new JAL statements, authorities can be associated with individual jobs. These authorities determine which users or groups of users are allowed to display detailed information. Authorities can be based on RACF, ACF2, Top Secret, or TSO userids, and can also cause a prompt for users to match a specified string before allowing access.

Two levels of ThruPut Manager information are therefore available through UDF:

- General information is available to any user:
    - ◦ The ThruPut Manager services that currently control the job, such as Job Binding Services.
    - ◦ Whether the job is being held by ThruPut Manager.

- Detailed information can be displayed only by those users permitted to do so by the job's authorities:
  - Information associated with the job during JAL processing.
  - Volume information.
  - Details about ThruPut Manager services that currently control the job, for example Job Setup Services.

If not restricted by JAL, the access allowed by UDF depends on the method used to select the job. See the discussion later in this chapter.

Extensions to JAL to Support UDF

New JAL statements are available to support UDF. They are summarized here:

Statement	Description
JOB_DATADEF	Define information for later display by operators or ISPF users.
JOB STORE	Associates data defined by prior JOB_DATADEF statements with the job currently being processed by ThruPut Manager.
JOB USER_DISPLAY	Provides authorization information to the User Display Facility.

These new statements can be applied to each job processed by ThruPut Manager:

- The JOB_DATADEF and JOB STORE statements make job-related information available to an ISPF user. Any data from JAL Job Descriptors can be passed to the ThruPut Manager User Display Facility. The JOB STORE Action Statement then allows you to specify the intended use for data created with JOB_DATADEF.
- The JOB USER_DISPLAY statement provides a mechanism to restrict the display of job-related information to specific ISPF users.

Information collected during ThruPut Manager processing is associated with individual jobs. To do this, you simply use the JOB_DATADEF statement to combine text with the Job Descriptors for the job. Alternatively, you can define messages that communicate decisions made in JAL that affect the job. For example:

```
JOB_DATADEF   TM_INFO_L1  ('Requires ',$UNITS_CART_ALL,' Tape Units')
JOB_DATADEF   TM_INFO_L2  ('Testing job, assigned Overnight Service')
```

The information is made available to UDF with the JOB STORE statement through the USER_DATA keyword:

```
JOB STORE USER_DATA(LINE1(TM_INFO_L1),LINE2(TM_INFO_L2))
```

The information, however, is not necessarily available to all users. Also associated with a job are authorities that can be set during ThruPut Manager processing using the JOB USER_DISPLAY statement:

```
JOB USER_DISPLAY USERID(ACC3*)
```

This statement assigns authorities to the job that allow only userids beginning with the characters ACC3 to examine detailed information.

For complete syntax descriptions and capability of the UDF JAL statements, see the ***JAL Reference Guide*** and the ***DAL/JAL User Guide***.

Controlling Access to Information

The ability to display detailed job information can be restricted. The JAL statement JOB USER_DISPLAY allows you to choose the restrictions you want to apply to the display of detailed job information:

- You can use JAL pattern matching to authorize specific users for individual jobs.
- You can elect to pass information to the same exit your installation uses to process the TSO STATUS command.

Note that if in your JAL you code:

```
JOB USER_DISPLAY USERID(*)
```

then the installation exit used to process the TSO STATUS command is ***not*** called, and all ISPF users are authorized to view job-related data. This is the default.

Using Your Installation's Exit

If you code the EXIT form of the JOB USER_DISPLAY statement, the installation exit used to process the TSO STATUS command is assumed to perform all authorization checking. This is indicated in JAL by coding:

```
JOB USER_DISPLAY EXIT
```

For a syntax description of the JOB USER_DISPLAY EXIT statement, see the ***JAL Reference Guide***.

When the JOB USER_DISPLAY statement is processed, authorization information about the job is stored for use by UDF. When a user attempts to use UDF, it calls the exit routine your installation uses to process the TSO STATUS command.

The exit routine as supplied by IBM is called IKJEFF53. It is invoked in the manner documented in the manual ***SPL: TSO Extensions User Exits and Modifications Volume 2***, except that registers 2 and 3 contain information identifying the call as an UDF request.

Registers At Entry

The following describes the expected contents of the registers at entry to your installation's exit.

Register 1

Register 1 contains the address of the standard ten-word parameter list.

To format the parameter list pointed to by register 1, you can code the following IKJEFFIE mapping macro instruction:

```
IKJEFFIE  IETYPE=CANST
```

This generates a parameter list as shown in the table below.

WORD	CONTENTS
1	Address of the jobname.
2	Address of a halfword containing the length of the jobname.
3	Address of the userid.
4	Address of one byte containing the length of the userid.
5	This field is set to zeros.
6	This field is set to zeros.
7	Address of one byte containing a command code of zero (0), indicating a **STATUS**-type request.
8	Address of the JES2 jobid.
9	Address of a halfword containing the length of the jobid.
10	This field is set to zeros, except that the high-order bit is turned on to indicate the end of the parameter list.

Register 2

Register 2 contains the character string ‘TMIN’. This distinguishes a UDF request from that of a STATUS or CANCEL.

Register 3

Register 3 contains the address of the DATA string provided by JAL. This register contains zeros if no DATA string was provided. The format of this area is:

```
llstring
```

Where:

`ll`

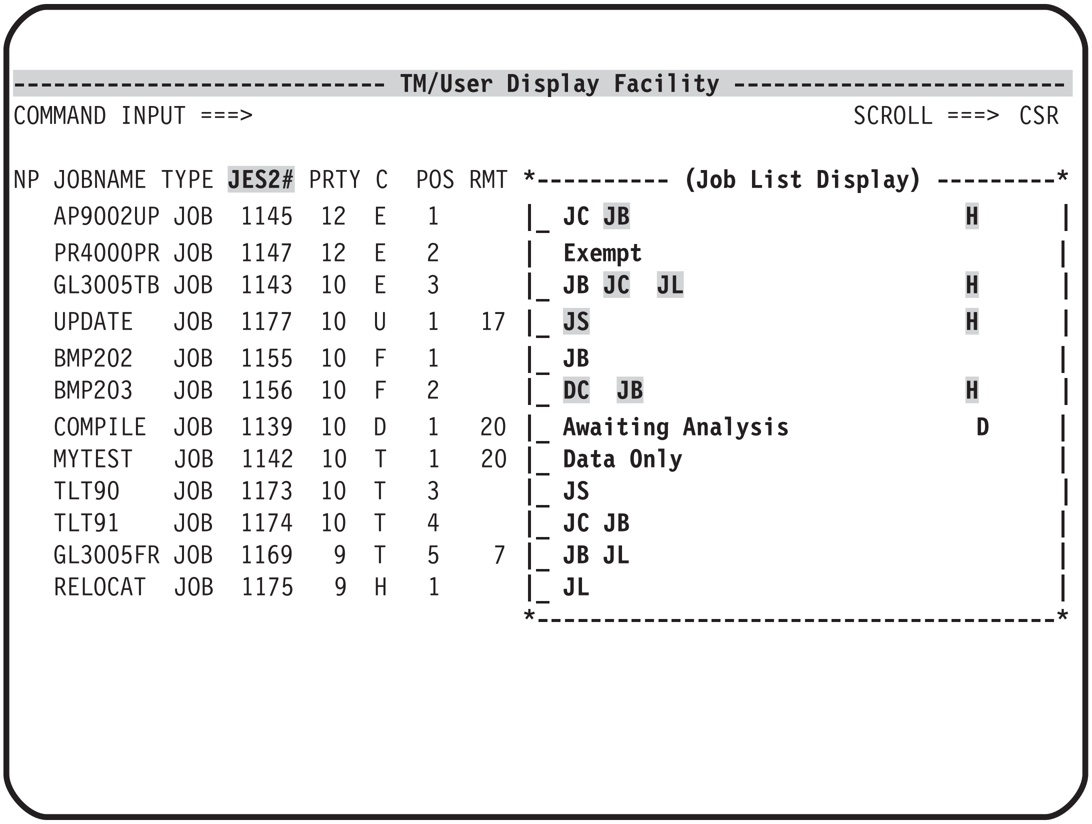
Is a two-byte field containing the length of the entire string (including the length field).

`string`

Is a maximum of 255 bytes of your data. The length field does not count towards this maximum.

TM/UDF and Variable Spool Manager Headers

Some spool managers, such as CA/Sysview, allow you to vary the information in the headers used in their displays. It is also possible that these headings can vary from release to release of a product. Because TM/UDF uses this header information to locate the JES2 job numbers on the screen, it might be necessary to indicate to ThruPut Manager what the possible headings are for your installation. This is the purpose of the UDF SET initialization statement.



```
---------------------------- TM/User Display Facility -------------------------
COMMAND INPUT ===>                                             SCROLL ===>  CSR

NP JOBNAME TYPE JES2# PRTY C  POS RMT *---------- (Job List Display) ---------*
   AP9002UP JOB  1145  12  E   1      |_ JC JB                          H      |
   PR4000PR JOB  1147  12  E   2      |  Exempt                                |
   GL3005TB JOB  1143  10  E   3      |_ JB JC  JL                      H      |
   UPDATE   JOB  1177  10  U   1   17 |_ JS                             H      |
   BMP202   JOB  1155  10  F   1      |_ JB                                    |
   BMP203   JOB  1156  10  F   2      |_ DC  JB                         H      |
   COMPILE  JOB  1139  10  D   1   20 |_ Awaiting Analysis               D     |
   MYTEST   JOB  1142  10  T   1   20 |_ Data Only                             |
   TLT90    JOB  1173  10  T   3      |_ JS                                    |
   TLT91    JOB  1174  10  T   4      |_ JC JB                                 |
   GL3005FR JOB  1169   9  T   5    7 |_ JB JL                                 |
   RELOCAT  JOB  1175   9  H   1      |_ JL                                    |
                                      *----------------------------------------*
```

Example of altered heading for SDSF used with UDF

In the screen above, the highlighted portion of the header line:

```
JES2#
```

is an example of a header that has been altered. To inform ThruPut Manager of this change so that TM/UDF will work properly, you use the UDF SET initialization statement:

```
UDF SET SDSF_JOBID(JES2#)
```

Details of the syntax and usage of this statement are found in “Chapter 2 . TMSS Initialization Statements.”

Implementation Summary

UDF can be added to an ISPF environment quickly and easily. UDF uses standard ISPF interfaces and requires no modifications to the system. UDF has the following “parts”:

- TMUSER, a ISPF command to invoke UDF.
- A set of ISPF panels.
- ISPF HELP entries.
- A set of ISPF messages.
- A command table for TMUSER subcommands.

Details concerning the installation of UDF can be found in the ***Installation Guide***

Dependencies

The only prerequisites for UDF are:

- ISPF Version 2 Release 1 or later, and
- ThruPut Manager Version 4 or later.

Chapter 6. ThruPut Manager ISPF Services (TMISPF)

This chapter provides a short introduction to the ISPF services provided to support ThruPut Manager.

Introduction

The ThruPut Manager base product includes an ISPF framework that allows various ThruPut Manager components and services to take advantage of interactive displays and dialogs. The services provided through ISPF are designed to make certain systems programming and operating tasks easier to manage.

☞ ***In order to use this facility, the appropriate ISPF datasets must be concatenated during the INSTALL process described in the Installation Guide.***

Some of the supported functions apply to optional components, therefore they might not be functional at your installation. To see exactly what services are available at your installation, invoke the TMISPF command:

```
TMISPF
```

The following pages illustrate the selection menus for the services so that you can see what is available. Full documentation for all ThruPut Manager ISPF services is provided through online help.

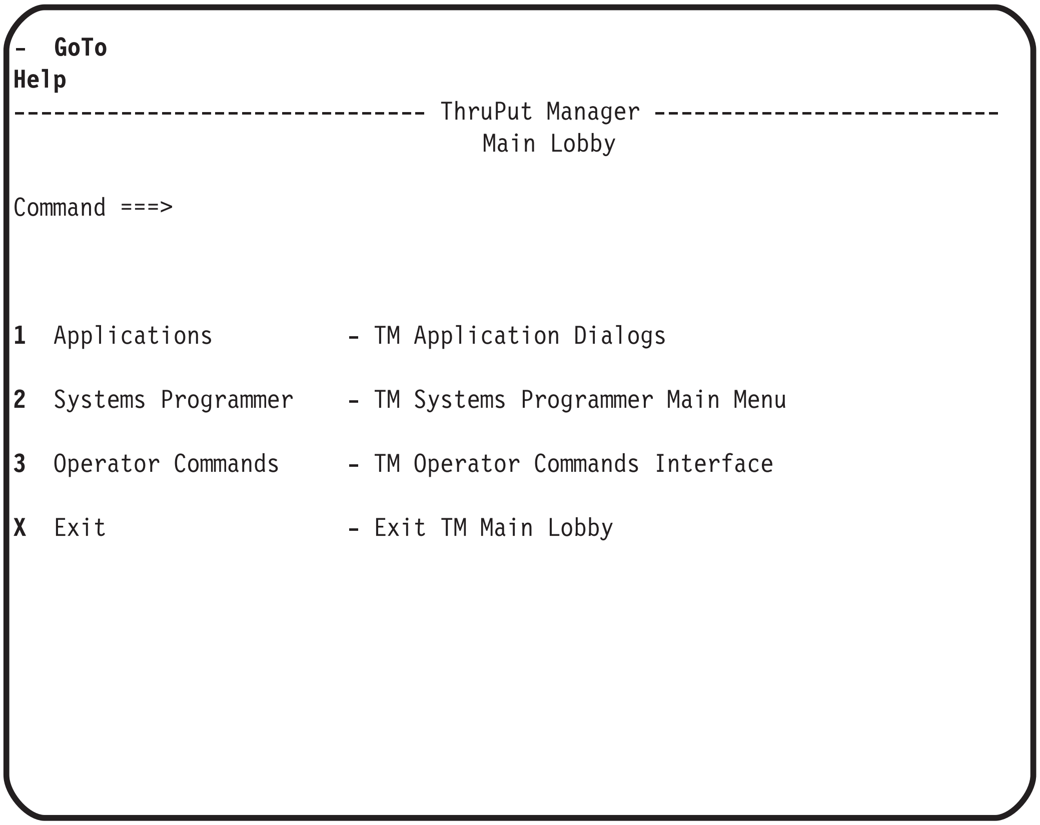
Note: TMISPF uses TSO Extended Consoles to provide some of its functions. This means that TSO sessions that use TMISPF must have the correct authority, since the Extended Console has the authority of the TSO session. To use ThruPut Manger operator commands through TMISPF, the TSO session must have SYS authority. To display ThruPut Manager information, the session must have INFO authority.

If the System Authorization Facility (SAF) for Operator commands is active the Console Authority is not used. Access to the commands is protected under the **OPERCMDS** class by a Resource of $$TM.function.command. For more details see *Authorizing the Use of Operating Commands* in "Chapter 5. Functional Services" of the ***Operating Guide***.

The Main Lobby

The TMISPF command takes you to the Main Lobby, which is the starting point for all ThruPut Manager ISPF services. The screen shown here illustrates the contents of the TMISPF Main Lobby.

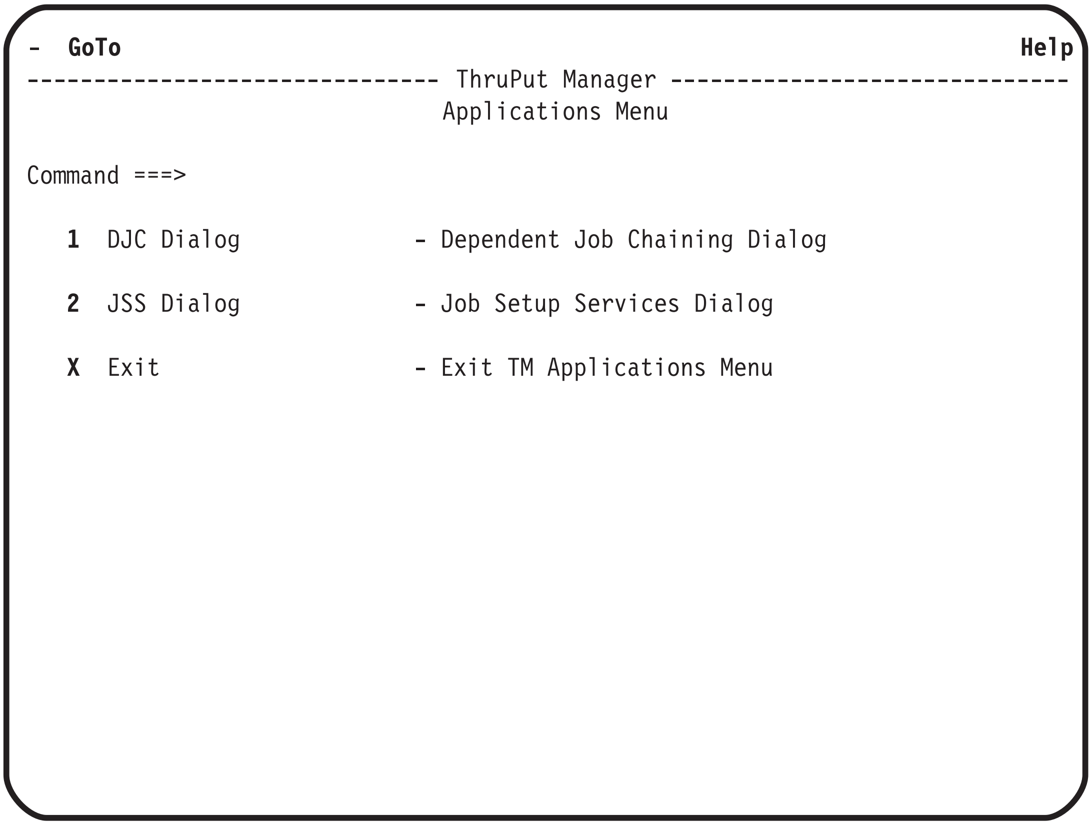
Each selection in the Main Lobby takes you to a list of services designed to supplement a particular set of ThruPut Manager functions.



```
-  GoTo
Help
------------------------------- ThruPut Manager --------------------------
                                   Main Lobby

Command ===>



1  Applications          - TM Application Dialogs

2  Systems Programmer    - TM Systems Programmer Main Menu

3  Operator Commands     - TM Operator Commands Interface

X  Exit                  - Exit TM Main Lobby
```

The TMISPF Main Lobby

Applications

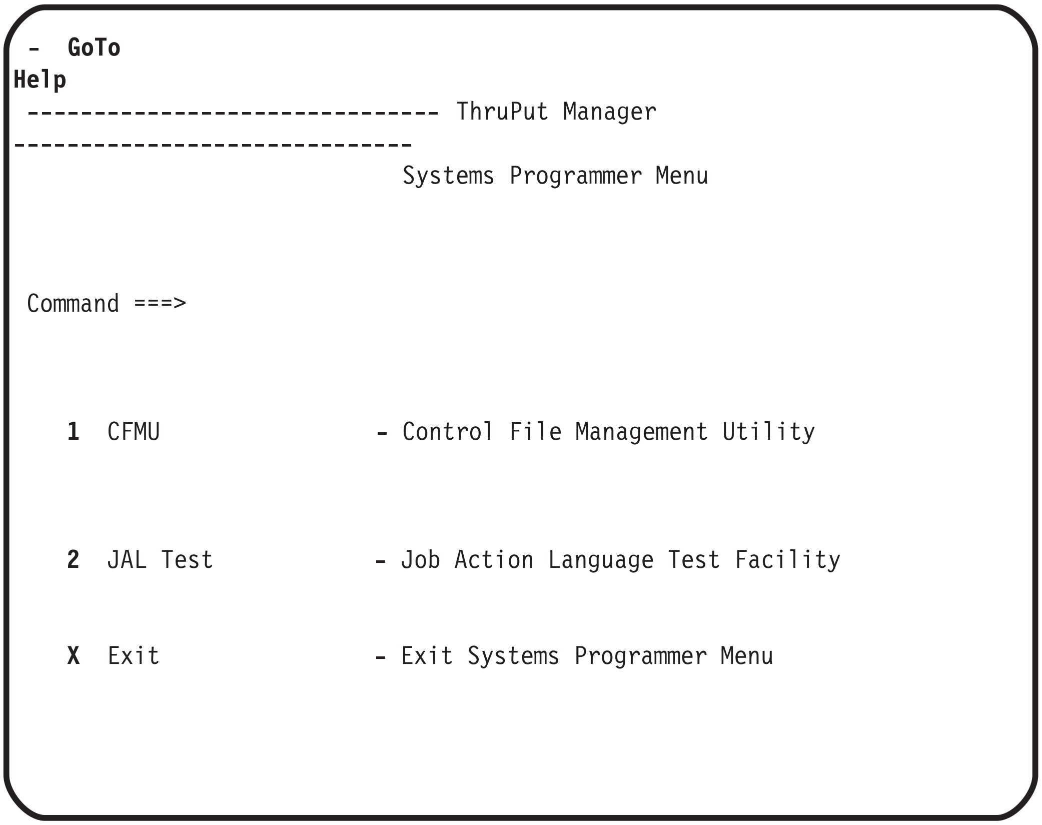


```
-  GoTo                                                                       Help
------------------------------ ThruPut Manager ------------------------------
                               Applications Menu

Command ===>

   1  DJC Dialog            - Dependent Job Chaining Dialog

   2  JSS Dialog            - Job Setup Services Dialog

   X  Exit                  - Exit TM Applications Menu
```

The Application Menu

This screen shows the Applications menu, which provides an ISPF interface to dialogs to manage ThruPut Manager applications. Note that you can select only applications that are associated with components that are installed at your installation.

The Systems Programming Services Menu

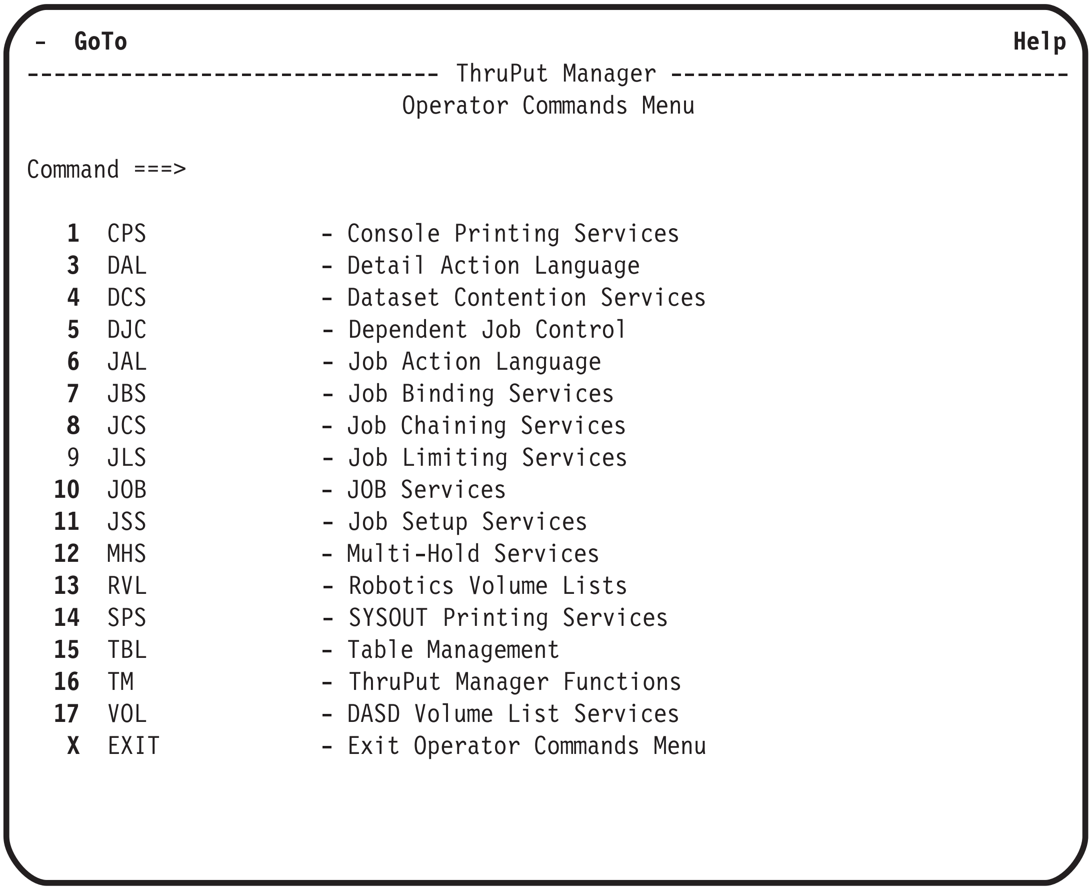


```
 -  GoTo
Help
 ------------------------------- ThruPut Manager
------------------------------
                             Systems Programmer Menu



 Command ===>



    1  CFMU               - Control File Management Utility



    2  JAL Test           - Job Action Language Test Facility


    X  Exit               - Exit Systems Programmer Menu
```

The Systems Programming Services Menu

This screen illustrates the Systems Programming Services menu, which provides an ISPF interface to ThruPut Manager tools designed for those who install and tune ThruPut Manager, and those designing and writing DAL and JAL.

The Operating Services Menu



```
 -  GoTo                                                                 Help
------------------------------ ThruPut Manager ------------------------------
                            Operator Commands Menu

Command ===>

   1  CPS              - Console Printing Services
   3  DAL              - Detail Action Language
   4  DCS              - Dataset Contention Services
   5  DJC              - Dependent Job Control
   6  JAL              - Job Action Language
   7  JBS              - Job Binding Services
   8  JCS              - Job Chaining Services
   9  JLS              - Job Limiting Services
  10  JOB              - JOB Services
  11  JSS              - Job Setup Services
  12  MHS              - Multi-Hold Services
  13  RVL              - Robotics Volume Lists
  14  SPS              - SYSOUT Printing Services
  15  TBL              - Table Management
  16  TM               - ThruPut Manager Functions
  17  VOL              - DASD Volume List Services
   X  EXIT             - Exit Operator Commands Menu
```

The Operating Services Menu

This screen illustrates the Operating Services menu, which provides an ISPF interface to panels that build operator commands for those concerned with controlling the system through console commands and those fetching and managing mountable volumes. Note that you can select only commands that are associated with components that are installed at your installation.

Implementation Summary

TMISPF can be added to an ISPF environment quickly and easily. TMISPF uses standard ISPF interfaces and requires no modifications to the system. TMISPF has the following elements:

- A set of ISPF panels.
- ISPF HELP entries.
- A set of ISPF messages.

Details concerning the installation of TM/ISPF can be found in the ***Installation Guide***.

Dependencies

The prerequisites for TM/ISPF are:

- ISPF Version 2 Release 1 or later, and
- ThruPut Manager Version 4 or later.

Chapter 7. JECL Services

The JECL functions available with ThruPut Manager.

JECL Services, including details of each JECL statement has been moved to the ***JECL Reference Guide***.

Chapter 8. Job Analysis Affinity

This chapter documents The Job Analysis Affinity facility. It allows an installation to control the affinity of a job for conversion and analysis purposes.

Description

The Job Analysis Affinity facility of ThruPut Manager allows an installation to specify a system, or systems, affinity default in a MAS complex for the CONVERSION and analysis process. The default is applicable to the jobs entering a system where this facility (AFFINITY) has been requested.

In a standard JES2 system, the default value is ANY and no facilities are provided to change it.

How to Request the Service

The ThruPut Manager JES2 statement TMPARM provides a keyword for this purpose. The syntax is:

```
AFFINITY=*
AFFINITY=ANY
AFFINITY=(sys1,sys2,...)
```

Where:

`*`

Indicates that the affinity is to be set for “this system”. That is, the system where the job was submitted.

`ANY`

This is the default, if this keyword is not used.

`(sys1,sys2,...)`

A list of valid JES2 system names. Up to seven can be specified.

You can also specify combinations of the above. For example:

```
AFFINITY=(*,sys6)
```

means “this system” and SYS6.

Notes:

- Once the job has completed analysis, the job affinity reverts to ANY.
- If the job is EXEMPT (no ThruPut Manager processing) the affinity reverts to ANY after conversion is done.
- If the job contains a /*JOBPARM statement that requests affinity, or if the RDR that processed the job is set to a specific affinity, their values override the AFFINITY value in TMPARM.

Considerations

In situations where jobs are to be converted, analyzed, and executed in the same system where they were submitted the following is necessary:

- Include the AFFINITY=* keyword in the TMPARM JES2 initialization statement (for each system).
- In JAL, you have to test for the system where JAL is running. This is best shown with an example:

```
IF ($JALSYS(SYS1))
   JBS SET SYSAFF(SYS1)
ORIF ($JALSYS(STSO))
   JBS SET SYSAFF(STSO)
OTHERWISE
   JBS SET SYSAFF(SIMS)
ENDIF
```

In this example there are three systems in the MAS complex, named SYS1, STSO, and SIMS. The above logic and action statements (combined with the use of AFFINITY=* in the TMPARM statement) ensures that the jobs are converted and can only be selected for execution in the system where they were submitted.

Chapter 9. Job Extract (JEF) Facility

This chapter describes the Job Extract Facility that allows you to obtain information collected by ThruPut Manager about the currently executing job.

The Facility

There are times when your installation wants information that ThruPut Manager has collected for a particular job during the Job Analysis phase. The Job Extract Facility provides services so you can request that information be returned to a program (the caller) about the job currently executing in the same address space as the caller. You can make a request from these environments:

- A user program.
- An SMF exit.
- A ThruPut Manager Job Analyzer post-JAL exit (exits 6, 7, 8, and 9).

You cannot make a request for a job running in a different address space, or a job that is not executing.

Three items are provided to support the facility:

- A module called DTMXTRCT. This is the service module that provides the logic to move the information to you.
- A macro called DTMXTRCT. This is the macro that you can use to invoke the above service module.
- A macro called DTMXFE. This is the macro provided to map the eXtractor Field Elements (XFEs) that are returned by the service module.

How JEF Works

To extract information, you simply code a list of ***character or range*** Job Descriptor names using the DTMXTRCT macro.

☞ ***The Job Extract Facility does not support Unique Job Descriptors.***

A special token, $ALL, is provided so that you can request all fields that are associated with character and range Job Descriptors. When you use this token, the XFEs are returned in alphabetical order by token name.

Each Job Descriptor name in the list describes a field to be extracted. For example, to extract the programmer name you include $PGMR in the list passed to DTMXTRCT. To extract the number of scratch volumes to be mounted on manual drives, include $SCRATCH_MANUAL.

Here are some other considerations:

- If the field represented by the name or token contains data (other than blanks or zeros) an XFE is constructed and returned for that token.
- If a character field contains blanks or is null), an XFE ***is not*** returned.
- If multiple occurrences of the field exist, one XFE is returned for each occurrence of the data.
- XFEs are returned in the same order in which they were requested (list passed to DTMXTRCT).

Considerations

The following notes should be considered when processing any of the Descriptors listed in the table below.

JEF Considerations for Job Descriptors

Descriptor Name	Notes
$ACCT(nn)	A request of the form $ACCT(nn) returns the **nn**th sub-field of the account data. The identity of the XFE is the same as for a request without a subscript ($ACCT), but the data returned consists of the length and contents of only the requested sub-field.
$ALL	$ALL requests that the JEF return all tokens associated with Job Descriptors (those beginning with $).
$INSYSAFF	This is the 32 bit System Affinity Mask. Systems prior to JES2 5.1 return an 8 bit mask in the first byte.
$JBACT $JBDEACT	One Agent name per XFE is returned (if present). Each possible level (up to 2) is returned in an 8-character field left justified.
$JBBIND	Up to four Agent names per XFE could be returned. Each possible level of the Agent name (up to 2) is returned in an 8-character field left justified. There can be up to 24 occurrences from JECL.

JEF Considerations for Job Descriptors	
Descriptor Name	**Notes**
$USERCnn $USERNnn	Requests for these tokens return meaningful data only when issued from a post-JAL Job Analyzer exit.

Environment

This facility can be invoked by any routine allowed to issue an SVC (e.g. GETMAIN macro). Note that in order to issue a request for CURRSAF (current system affinity), ***the routine must be authorized.***

Description of Macros

The following pages give a detailed description of the macros provided with JEF. The syntax and notation follow ThruPut Manager conventions.

DTMXTRCT

Invokes the JEF Services Module

This macro is provided to simplify the invocation of the service module. It generates an inline parameter and list of names and/or tokens.

	DTMXTRCT	name-list[,SP={3 | nn | (register)}
		,RMODE={24 | 31 | (register)}
		,TYPE={LINK | CALL | (register)}
		,MF=I]

name-list

One or more Job Descriptor names, separated by commas.

Note that $ALL is a special token name to indicate you want all fields associated with Job Descriptors. If requested, the list is returned in alphabetical order.

SP={3 | nn | (register)}

The sub-pool ID for the XFEs to be constructed.

The default is sub-pool 3.

3 | nnn | (register)

The sub-pool number or a register number that is to contain the sub-pool number.

RMODE={24 | 31 | (register)}

The residency mode of the XFEs.

The default is 24 (below the line).

24 | 31 | (register)

Indicates that the XFEs are to be created below the line (24), above the line (31) or that a register is to contain the residency mode.

TYPE={LINK | CALL | (register)}

Indicates the type of linkage to be generated by the macro.

The default is LINK.

LINK | CALL | (register)

When LINK is specified, the LINK macro is used.

When CALL is specified BALR is used. ***Note that this implies that the service module DTMXTRCT is link-edited with your program. This technique is not recommended.***

When a register value is provided it means that BALR is to be used, but the address of the service module DTMXTRCT is in the specified register.

`MF=I`

Indicates that the parameter list and the list of names is to be expanded “inline”. This is the default.

If the register form of SP or RMODE is used, the expansion is not reentrant.

Register usage:

The SP and RMODE, when placed in a register, must be in the low order byte. Any register can be used except R1.

The TYPE parameter can specify any register except R1.

DTMXTRCT

Invokes the JEF Service Module

This is the List and Execute form of the DTMXTRCT macro. It generates remote parameter lists for tokens and call parameters.

LIST FORM:

parmlabel	DTMXTRCT	MF=L

tokenlabel	DTMXTRCT	token-list,MF=L

EXECUTE FORM:

	DTMXTRCT	{name-list | address-of-name-list} [,SP={3 | nn | (register)}] [,RMODE={24 | 31 | (register)}] [,TYPE={LINK | CALL | (register)}] ,MF=(E,parm-list-address | (register))

List Form

`MF=L`

When coded with MF=L only, the expansion generates a ***remote parameter list.***

Note that the **parmlabel** ***represents the address of the remote parameter list.***

`name-list,MF=L`

The expansion of this format generates a *remote list of names*.

Note that the **tokenlabel** ***represents the address of the remote list of names.***

Execute Form

`Token-list | address-of-name-list`

The execute form of the macro allows you to generate an inline list of names, or you can provide the address of the remote list.

`SP={3 | nnn | (register)`

The sub-pool ID for the XFEs to be constructed.

The default is sub-pool 3.

`3 | nnn | (register)`

The sub-pool number or a register number that is to contain the sub-pool number.

`RMODE={24 | 31 | (register)`

The residency mode of the XFEs.

The default is 24 (below the line).

`24 | 31 | (register)`

Indicates that the XFEs are to be created below the line (24), above the line (31) or that a register is to contain the residency mode.

`TYPE={LINK | CALL | (register)`

Indicates the type of linkage to be generated by the macro.

The default is LINK.

`LINK | CALL | (register)`

When LINK is specified, the LINK macro is used.

When CALL is specified BALR is used. ***Note that this implies that the service module DTMXTRCT is link-edited with your program. This technique is not recommended.***

When a register value is provided it means that BALR is to be used, but the address of the service module DTMXTRCT is in the specified register.

`MF=(E,{parm-list-address | (register)})`

For the execute form of the macro you must supply either the address of the remote parameter list or a register that will contain that address at execution time.

If you want the execute form of the macro you must specify this parameter.

Register usage:

The SP and RMODE, when placed in a register, must be in the low order byte. Any register can be used except R1, R14, and R15.

The TYPE parameter can specify any register except R1.

The MF=E parameter can specify any register; however, R1 produces a smaller expansion.

The DTMXTRCT Service Module

DTMXTRCT is link-edited with an RMODE of ANY.

Your program must be using a 31 bit addressing mode if it gets the address of DTMXTRCT from the LOAD macro, and invokes it with TYPE=(register).

When DTMXTRCT returns control:

- R1 contains the address of the XFE list or 0.
- R15 contains one of the following return codes:

  0 Your list of names was all correct and ThruPut Manager control blocks were available.

  4 Your list of names contained at least one unrecognizable name. All ThruPut Manager control blocks were available.

  8 All ThruPut Manager control blocks were not available.

In all cases, check R1 to see if it contains the address of any XFE.

XFE Mappings

The XFEs returned are mapped by the DTMXFE macro, as shown here:

```
XFE    DSECT
XFETOKNDS     CL8 DESCRIPTOR/TOKEN NAME
         ORG        XFETOKN
XFETKPFX   DS         CL6              TOKEN PREFIX (1ST 6 OF 24)
XFETKID#   DS         H                TOKEN ID NUMBER (UNIQUE)
XFEINDXDS     AL1 NTH ENTRY IN TOKEN LIST
**NOTE**                    XFEINDX IS 0 IF > 255 TOKENS IN LIST
XFESPIDDS     AL1 SUBPOOL ID
XFEGLENDS     H   XFE LENGTH (WITH  PREFIX)
XFENEXTDS     A   NEXT XFE ADDRESS OR ZERO
XFEDATADS     OD  START OF DATA
*
XFELEN EQU    *-XFE    LENGTH OF XFE PREFIX
XFE_$ACCT             EQU 256*C'A'+1
XFE_$ACFLID           EQU 256*C'A'+2
...
```

XFE Mapping

Notes:

- You can determine the length of the data returned by subtracting the prefix length (XFELEN) from the total length value returned in XFEGLEN. The length and type of each Job Descriptor is listed in the ***JAL Reference Guide***.

- If any XFEs are returned, register 1 points to the first one. The address contained in XFENEXT points to the next XFE. The last one contains binary zeros.

- JEF returns the first six characters of the associated Job Descriptor name in the XFETKPFX field, and a numeric Token ID in the XFETKID# field. The DTMXFE macro maps numeric Token IDs. To refer to them, prefix the Job Descriptor name with ‘XFE_’ as shown below:

  ```
  XFE_descriptorname
  ```

  ☞ ***Because new Descriptors are often added, the DTMXFE macro is the definitive source for these numeric Token IDs.***

- XFEs are returned in the same order as requested in the list of names passed to DTMXTRCT. The XFEINDX field contains a one byte binary number that represents the order of the corresponding name in the list. That is, if a name is the third, the corresponding XFE contains a 3 in the XFEINDX field. This lets you construct branch tables that make the processing of XFEs easier.

  ☞ ***It is your responsibility to release the XFE storage when you no longer need it.***

- The SP parameter in the DTMXTRCT macro simplifies this task. By selecting a sub-pool that is not used for any other purpose in your program, you can release all the XFEs with a single call to FREEMAIN. The default of sub-pool 3 is probably not used anywhere else in your program. Note that the sub-pool number is contained in the XFESPID field of every XFE.

Extracting Data Automatically

ThruPut Manager supports automatic SMF data extraction with the TM SMF initialization statement. For details, refer to “Chapter 10. SMF Data Collection for Job Analysis.”

Examples

Example 1

A user program extracts the following fields:

- Account Field.
- Programmer Name Field.
- Room Number Field

The program uses the simplest form of the DTMXTRCT macro.

The required assembler code is shown below:

```
         ...
         DTMXTRCT ($ACCT,$PGMR,$ROOM)
         LTR      R9,R1                   ANY XFES?
         BZ       NOXFES                  NO
         USING    XFE,R9                  MAP XFES
         CLI      XFETOKN+1,C'A'          ACCOUNT FIELD?
         BNE      CHKPGMR                 NO
         MVC      RECORD1(20),XFEDATA     SAVE IT
         ICM      R9,15,XFENEXT           LOAD NEXT XFE
         BZ       FREEXFES                NO MORE
CHKPGMR  CLI      XFETOKN+1,C'P'          PGMR NAME?
         BNE      CHKROOM                 NO
         MVC      RECORD2(20),XFEDATA     SAVE IT
         ICM      R9,15,XFENEXT           LOAD NEXT XFE
         BZ       FREEXFES                NO MORE
CHKROOM  CLI      XFETOKN+1,C'R'          ROOM NUMBER?
         BNE      FREEXFES                NO
         MVC      RECORD3(4),XFEDATA      SAVE IT
FREEXFES FREEMAIN RU,SP=3
NOXFES   DS       OH
         ...
         ...
         DTMXFE                           MAP XFES
```

Example 1. Simple Form of DTMXTRCT Macro

Example 2

In this example we show the use of the LIST and EXECUTE forms of the DTMXTRCT macro.

Also the handling of multiple occurrences of a field is shown.

The user requests four fields:

- $JOBNAME
- $RACFU
- $JBBIND
- $JLS_LIMIT

The first two fields are single occurrence fields.

The next two fields are multiple occurrence fields:

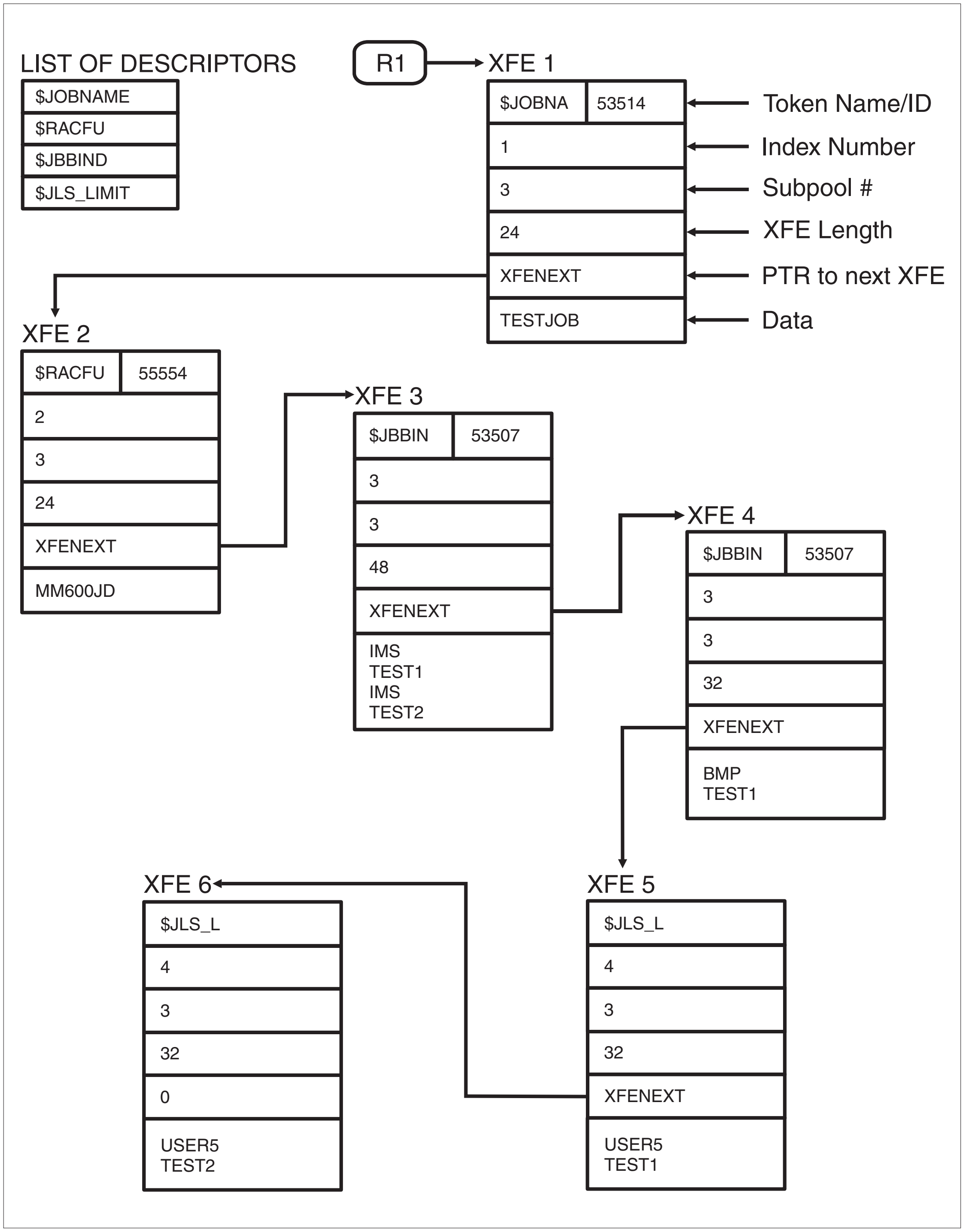
- In the case of $JBBIND there can be up to 96 occurrences of the field: 24 each from JECL control statements, JAL, DAL, and SAC. Each occurrence can have from one to four Agents.
- In the case of $JLS_LIMIT there can be up to 24 occurrences. Only one Agent per occurrence is possible.

☞ ***Each Binding and Limiting Agent is presented as two 8-byte fields. The actual characters are left justified padded with blanks. If the Agent is a single level name Agent, the second field contains blanks.***

The next page provides a diagram showing the structure of the XFEs that are returned. A total of 6 XFEs are returned. In this example:

- The job name is TESTJOB.
- The RACF userid is MM600JD.
- There are two JBS BIND statements. The first one contains two Agents named IMS.TEST1 and IMS.TEST2. The second one contains one Agent named BMP.TEST1.
- There are two Limiting Agents. The first one has a name of USER5.TEST1 and the second one USER5.TEST2.

Note that the index (3) for XFE3 and XFE4 is the same. So is the index (4) for XFE5 and XFE6.



Structure of XFEs

The assembler code needed to process the situation described above is shown in the next two figures:

```
         ...
         LA      R7,SMFBIND              SET OUTPUT ADDRESS FOR BIND AGENTS
         LA      R8,SMFLIM               SET OUTPUT ADDRESS FOR LIMIT AGENTS
*
         DTMXTRCT TOKLST,MF=(E,XTRLST)   EXECUTE FORM OF MACRO
         LTR     R9,R1                   ANY XFES?
         BZ      NOXFES                  BRANCH IF NO XFES
         USING   XFE,R9                  SET ADDRESSABILITY
LOOP     SLR     R15,R15
         IC      R15,XFEINDX             PICK TOKEN NUMBER
         SLL     R15,2                   PREPARE IT TO USE FOR BRANCH TABLE
         B       *(R15)                  BRANCH TO BRANCH TABLE
*
*        BRANCH TABLE
*
         B       JOBNAME                 BRANCH TO PROCESS TOKEN INDEX 1
         B       RACFU                   BRANCH TO PROCESS TOKEN INDEX 2
         B       JBSBIND                 BRANCH TO PROCESS TOKEN INDEX 3
         B       JLSLIM                  BRANCH TO PROCESS TOKEN INDEX 4
*
NXTXFE   ICM     R9,15,XFENEXT           GET PTR TO NEXT XFE
         BNZ     LOOP                    IF NOT ZERO REPEAT PROCESS
         FREEMAIN RU,SP=3                NO MORE XFES - FREE STORAGE
NOXFES   DS      0H
         ...

JOBNAME  MVC     SMFJNAME,XFEDATA        MOVE JOB NAME
         B       NXTXFE
RACFU    MVC     SMFRACFU,XFEDATA        MOVE RACFU
         B       NXTXFE
JBSBIND  LH      R15,XFEGLEN             CALCULATE DATA LENGTH
         SH      R15,=Y(XFELEN)
         BCTR    R15,0                   SET LENGTH FOR EX MVC
         EX      R15,BINDMVC
         LA      R7,1(R15,R7)            SET OUTPUT POINTER FOR
         B       NXTXFE                  NEXT POSSIBLE JBSBIND XFE
```

Example 2. List and Execute Form of DTMXTRCT (1 of 2)

```
*
BINDMVC  MVC     0(0,R7),XFEDATA EXECUTE JBSBIND MOVE
*
JLSLIM    MVC     0(16,R8),XFEDATA        MOVE LIMITING AGENT
         LA      R8,16(,R8)              SET OUTPUT PTR FOR NEXT
         B       NXTXFE                  POSSIBLE LIMITING XFE
         ...
*
*        CONSTRUCT REMOTE LIST OF DESCRIPTOR NAMES
*
TOKLST   DTMXTRCT ($JOBNAME,$RACFU,$JBBIND,$JLS_LIMIT),MF=L
         ...
         ...

WORK     DSECT
*
*        AREA FOR REMOTE PARAMETER LIST
*
XTRLST   DTMXTRCT MF=L
         ...
         ...
*
*        OUTPUT AREA
*
SMFREC   DSECT
         ...
SMFJNAME         DS      CL8             AREA FOR 8-CHARACTER JOB NAME
SMFRACFU         DS      CL7             AREA FOR 7-CHARACTER RACFU
SMFPAD1 DS       CL1                     ALIGNMENT
SMFBIND DS       48CL16                  AREA FOR UP TO 48 AGENTS
SMFLIM  DS       24CL16                  AREA FOR UP TO 24 AGENTS
         ...
*
*        MAP XFES
*
         DTMXFE
```

Example 2. List and Execute Form of DTMXTRCT (2 of 2)

Chapter 10. SMF Data Collection for Job Analysis

This chapter describes the SMF data collection facilities provided by ThruPut Manager to monitor activity during Job Analysis.

SMF Monitoring of ThruPut Manager Activity

A ThruPut Manager facility is provided for an installation to gather statistical information about ThruPut Manager's activity while analyzing jobs. An SMF record containing this information is generated for each job.

The data collected includes:

- CPU time used to interpret job.
- CPU time used to analyze job.
- Number of catalog LOCATEs.
- Number of DD statements.
- Number of DASD volumes and tape volumes.

Activating Data Collection

JEF data can be collected and made available automatically. To activate SMF data collection, you must include an initialization statement:

```
TM SMF TYPE(smf-record) [ANALYZER | ANALYZER(table-id[,subtable-id])] [DBS][DCS][SLM]
```

smf-record

Is a number from the range 128 to 255, representing the SMF record number to be used.

ANALYZER

Indicates that the Job Analyzer is to collect the default SMF data. See the following table *ThruPut Manager SMF Performance Record*.

ANALYZER(table-id[,subtable-id])

Indicates that the Job Analyzer is to collect data in addition to the default data.

`table-id`

Is a number from the range 1-9, identifying the ThruPut Manager table containing the desired Token names for the additional data.

`subtable-id`

Is an optional 1-24 character name identifying the portion of the ThruPut Manager table that contains the desired Token names.

`DBS`

Indicates that the DBS application is to collect SMF data. Refer to the ***Drive Booking Services: System Programming Guide***.

`DCS`

Indicates that the DCS application is to collect SMF data. Refer to the ***Dataset Contention Services: System Programming Guide***.

`SLM`

Indicates that SLM data is to be recorded. For further details, refer to the ***Setup Guide to ThruPut Manager AE Essentials***.

The mapping macro DTMSMFPR is provided to assist you in analyzing the collected data.

Summary of ThruPut Manager SMF Record

The following table is intended for use as a guide only. If there is a discrepancy between this summary and the mapping macro DTMSMFPR, the macro is correct. ***This table reflects only the data collected by default.***

THRUPUT MANAGER SMF PERFORMANCE RECORD			
For specific mapping, refer to macro DTMSMFPR			
Name	**Type**	**Length**	**Description**
SMFTMLEN	binary	2	Record length
SMFTMSEG	binary	2	Segment descriptor
SMFTMFLG	binary	1	System indicator
SMFTMRTY	binary	1	Record type

THRUPUT MANAGER SMF PERFORMANCE RECORD			
For specific mapping, refer to macro DTMSMFPR			
Name	**Type**	**Length**	**Description**
SMFTMTME	binary	4	Time since midnight (in 100ths of seconds) record was sent to SMF writer
SMFTMDTE	packed	4	Date (in the form yyddd) record was sent to SMF writer
SMFTMSID	char	4	System identification
SMFTMWID	char	4	Subsystem identifier
SMFTMSTP	binary	2	Record subtype: 15 for ThruPut Manager 6 for TM/DCS
SMFTMJBN	char	8	Job name
SMFTMRST	binary	4	Reader start time (in 100ths of seconds)
SMFTMRSD	packed	4	Reader start date (yyddd)
SMFTMJNM	char	8	JES job identifier
SMFTMINT	binary	4	CPU time used by interpreter and Job Analyzer initiation
SMFTMJAT	binary	4	Total CPU time used by Job Analyzer
SMFTMCTA	binary	4	Number of catalogs allocated during Job Analysis
SMFTMLRQ	binary	4	Number of catalog locates required
SMFTMLCM	binary	4	Number of catalog locates using catalog management
SMFTMLCA	binary	4	Number of catalog locates using catalog lookaside
SMFTMNDD	binary	4	Number of DD statements
SMFTMNDA	binary	4	Number of DASD volumes

THRUPUT MANAGER SMF PERFORMANCE RECORD			
For specific mapping, refer to macro DTMSMFPR			
Name	**Type**	**Length**	**Description**
SMFTMTC	binary	4	Number of tape cartridge volumes
SMFTMJST	char	4	Job status (OKAY, FAIL, or FLSH)
SMFTMACS	char	1	ACS support indicator (Y for active, N for inactive)
SMFXFENO	binary	4	Number of XFE extensions
SMFTMELP	binary	8	Elapsed time in Analyzer
SMFTMNOM	binary	1	Number of this record
SMFTMNON	binary	1	Total number of records

If the field SMFTMNON is greater than 1, there is more than one SMF record for this job. If the records are sorted in ascending order by SMFTMNOM, the last record has been processed when SMFTMNOM equals SMGTMNON (i.e. ***m*** of ***n*** records).

Record Subtypes

Some ThruPut Manager applications allow you to collect SMF data specific to the application. The primary ThruPut Manager SMF record mapped by DTMSMFPR contains a subtype indicator in the field SMFTMSTP:

- 15 indicates a ThruPut Manager Analyzer record.
- 6 indicates a TM/DCS record. For details about the use of this subtypes, refer to the ***Dataset Contention Services: System Programming Guide***.

Collecting More Than Default JEF Data

The TM SMF initialization statement allows you to collect more than the default JEF data. The additional data is added to the SMF record in XFE format, ***except that the field XFENEXT is an offset, not an address.*** The offset is relative to the start of the SMF record, including the RDW.

To extract the additional data, you must create a ThruPut Manager table containing the tokens you wish to extract. ThruPut Manager tables are described in "Chapter 15. Table (TBL) Function."

The table is defined in the usual manner, ***except that any Token name containing the underscore (_) character must be enclosed in quotes.***

Here is a sample table:

```
SUBTABLE(SMFDATA)
   $JOBNAME,$INCLASS
   $INSYSID
   JLSLIMIT
   '$UNIT_CART_ALL'
```

Sample Table for JEF Data Extraction

If this table were called table 9 to match the SMF exit number, the load statement could look like this:

```
TBL LOAD 9 TM.PARMLIB(TMSMFTBL)
```

Now modify the TM SMF initialization statement to point to the table:

```
TM SMF TYPE(255) ANALYZER(9,SMFDATA)
```

The XFEs are appended to the end of the standard ThruPut Manager SMF data. The DTMSMFPR macro has been updated to reflect this.

Chapter 11. Installation Exits Summary

This chapter summarizes the exit points available. For complete information about the designing, coding, and implementation of installation exits, see the manual ***Exits: System Programming Guide****.*

Introduction

ThruPut Manager provides a comprehensive set of installation exits, enabling you to accommodate virtually any special circumstances. Interfaces to accounting, security, and media library systems can be implemented easily through these facilities. Also, in the unlikely event that JAL cannot handle all your requirements for batch jobs, the exits provide the ability to incorporate special processing.

Key to the effective use of ThruPut Manager Exits are two facilities:

1. The User Descriptors for DAL and JAL. You can set values in any of the pre-JAL exits that can then be tested in your DAL or JAL logic to make decisions about any particular job. Character string and Numeric User Descriptors are provided.

   For a detailed explanation of the User Descriptors, refer to the publications ***DAL/JAL User Guide***, ***DAL Reference Guide***, and ***JAL Reference Guide***.

2. The ability to provide up to 255 characters of data at the time each exit is activated. That allows you to externalize information that normally varies. In that way, you only reassemble the exits when you want to alter the logic.

   For a detailed explanation, refer to the publication ***Exits: System Programming Guide.***

Job Analyzer Exits

Installation exits in the Job Analyzer environment share several characteristics:

- They are reLOADed for each new job processed by a Job Analyzer.
- Exits 2, 3, 4, 5, and 19 can be called more than once, depending on the nature of the job.
- Exits 2, 3, 4, and 5 are not called if the job is flushed by an installation exit, or failed on a JCL error by Early Scan.

- Exits 2, 3, 4, 5 and 19 do not receive control in “numerical order”.

Job Analyzer Exits		
Pre-JAL Exits		
Exit #	Exit Name	Description
Exit 1	Job Statement	This exit receives control when ThruPut Manager has finished processing the JCL for a job, and is about to start analysis of the job.
Exit 2	EXEC Statement	This exit receives control for each occurrence of an executable step in the job (every EXEC statement).
Exit 3	DD Statement	The exit receives control for each occurrence of a DD statement in the job.
Exit 4	Unit Name/Device Type	This exit receives control each time the ThruPut Manager Job Analyzer encounters a unique unit name, device type, or device number, and must decide how to classify the name, number, or type for Job Analysis purposes.
Exit 5	Volume Status	This exit receives control each time the ThruPut Manager Job Analyzer encounters a unique volume serial number. After an initial classification, the exit is invoked. The exit gives you an opportunity to alter the resource classification.(e.g., tape, DASD, MSS, mountable, unavailable) for Job Analysis purposes. Note that this exit is also invoked when a catalog is to be allocated on a volume.

Job Analyzer Exits		
Post-JAL Exits		
Exit #	Exit Name	Description
Exit 6	JAL Action Modification	This exit receives control after the ThruPut Manager Job Analyzer has classified a job using the JAL supplied by the installation.

Job Analyzer Exits		
Post-JAL Exits		
Exit #	**Exit Name**	**Description**
Exit 7	Job Information Collection	This exit receives control when ThruPut Manager is ready to save all the information collected about a job.
Exit 8	Requeue Message	This exit receives control for each requeue message inserted into the job's system messages dataset or sent to the operator.
Exit 9	Job Termination	This exit receives control when the ThruPut Manager Job Analyzer has completed processing of the job, and the job has been requeued.
Exit 19	JECL Statement inspection.	Allows you to inspect JECL statements that have been defined to ThruPut Manager.

TMSS Exits

Installation exits in the TMSS environment share the following characteristics:

- They are reLOADed only when TMSS is reinitialized or a task is reattached.
- They can be called more than once, depending on the nature of the exit.
- They do not all run under the same task, so the user data areas available in some exits are not available to all installation exits.

TMSS EXITS		
Exit #	**Exit Name**	**Description**
Exit 10	TMSS Initialization	This exit receives control when the TMSS address space is started.
Exit 11	TMSS Termination	This exit receives control when the operator requests that the TMSS address space terminate processing.
Exit 12	TMOS Command Inspection	This exit receives control when ThruPut Manager Operator Services (TMOS) has been given an operator command and is about to process it.

TMSS EXITS		
Exit #	**Exit Name**	**Description**
Exit 13	DISPLAY JOB	This exit receives control when ThruPut Manager Operator Services (TMOS) is processing a JOB DISPLAY command.
Exit 20	User Volume Key Add	This exit receives control whenever a volume serial number is added to the VIF.

Chapter 12. Multi-hold Services (MHS) Function

This chapter describes the Multi-hold Services (MHS) function.

Introduction

Multi-hold Services (MHS) is a ThruPut Manager Base Product function that provides support for holding or managing jobs by category and with optional annotation. Jobs can be held or *delayed*, meaning that MHS prevents a job from being selected for analysis or execution by removing its system affinity.

The holds or delay can be applied manually or automatically. The categories make it easier to determine why a job is being held or delayed. To complement this capability, MHS also supports the ability to add an annotation to the job, which can contain further information about why the hold or delay was applied.

The categories are:

- OPER, which are operator holds that are applied and removed with the /MHS_OPER HOLD and /MHS_OPER RELEASE commands.
- TM, which are holds that are applied automatically when ThruPut Manager must hold a job for reasons such as unavailable catalogs. These can be removed with the /MHS_TM RELEASE command.
- USER, which are holds applied either through the UHS (User Hold Services) dialog under ISPF or the /*MHS_USER HOLD JECL statement. Normally these holds are removed through the UHS dialog in ISPF, but they can also be removed with the /MHS_USER RELEASE command.
- VOL, which is a delay caused by removing the job's system affinity and thus making it unavailable for selection or execution. This occurs when ThruPut Manager determines that a DASD volume required by the job is unavailable. When the volume becomes available, the job automatically becomes eligible for selection. Additionally, the job can be made eligible for selection by using the /MHS_VOL RELEASE command.

These holds or delays are independent of both MVS/JES2 holds and other ThruPut Manager holds (JSS, JBS, etc.). Jobs can have one or more of these holds *and* other holds applied simultaneously.

MHS_OPER Holds

An MHS_OPER hold is a *manual* hold, applied through use of the /MHS_OPER HOLD command. Not only does the OPER category serve to indicate that the job has been deliberately held by an operator, but the NOTES keyword of the /MHS_OPER command supports up to 16 characters of explanatory information. This annotation is included in messages and displays, providing a way to document the reasons for an operator-applied hold. Additionally, jobs can be grouped through the ID keyword of the /MHS_OPER HOLD command. These groupings can be used to apply the /MHS_OPER DISPLAY and /MHS_OPER RELEASE commands to several jobs at once.

MHS_TM Holds

An MHS_TM hold is an *automatic* hold, applied by ThruPut Manager when specific conditions are detected during Job Analysis. These include:

- Catalog problems.
- Abends in ThruPut Manager services.

When ThruPut Manager encounters any of these conditions, the affected job is placed in MHS_TM hold, grouped by catalog or abend code, and a description of the problem is included as an annotation. The /MHS_TM DISPLAY and /MHS_TM RELEASE commands can use the catalog or abend code grouping to selectively display jobs and release them when the problems that caused the hold have been resolved.

USER Holds

TM/UCS ONLY

An MHS_USER hold is a *manual* hold that is applied by a user rather than an operator. When TM/UCS is installed, there are two methods users can employ to apply this category of hold:

- The UHS dialog under ISPF.
- The JECL statement /*MHS_USER HOLD.

The MHS_USER hold is intended to relieve operations from requests to hold jobs for users. It allows users to apply the hold and document it with a notation. Users also can remove the hold themselves, although if necessary the hold can be removed with the /MHS_USER RELEASE command.

VOL Delays

An MHS_VOL delay removes the job's system affinity, making it unavailable for analysis or execution. It is *automatic*, applied by ThruPut Manager when a DASD volume required by a job in execution is not available:

- NOT FOUND, indicating that ThruPut Manager could not find the volume on the system on which the job is running.
- OFFLINE, indicating that ThruPut Manager found the volume but it is offline and therefore not available.
- UNAVAILABLE, indicating that the volume is marked temporarily out of service, either by the /VOL ADD operator command or the VOL SET initialization statement.

Jobs in the MHS_VOL category are grouped by volume serial number, so that the /MHS_VOL DISPLAY and /MHS_VOL RELEASE commands can be applied to all jobs affected by a particular volume.

When the volume becomes available, ThruPut Manager automatically restores the system affinities for affected jobs.

Implementation Considerations

The MHS function is a part of the ThruPut Manager Base Product. There are no implementation procedures that are specific to MHS.

Your JAL can use the MHS_VOL SET EXEMPT statement to make a specific job ineligible for the MHS_VOL delay.

Note the way MHS handles catalog errors and unavailable volumes. Jobs encountering these problems are held or delayed automatically by MHS, together with an annotation providing details that can help resolve the problem. Operations procedures for these situations might therefore require review.

Normally, MHS_VOL delays are applied based on the results of a catalog search done at Job Analysis. If it is necessary to ensure this list is accurate when the job is selected for execution, you can use the SET CATALOG_LOOKUP(YES) statement in JAL. This forces the catalog search to be redone when the job is selected. Note that there are obvious performance consequences.

Dependencies

There are no dependencies for this function.

TMSS Initialization Statements

There are no TMSS initialization statements for this function.

Facilities Summary

MHS Operator Commands	
Command	**Description**
MHS DISPLAY	Displays MHS information for the job specified or for all jobs managed by MHS.
MHS RELEASE	Removes jobs from all MHS hold categories.
MHS_OPER DISPLAY	Displays MHS_OPER information for the job specified or for all jobs held by MHS for the OPERator category.
MHS_OPER HOLD	Applies an MHS HOLD to a job and places it in the OPERator category.
MHS_OPER RELEASE	Removes jobs from the MHS_OPER hold category.
MHS_TM DISPLAY	Displays MHS_TM information for the job specified or for all jobs held by MHS for the MHS_TM hold category.
MHS_TM RELEASE	Removes jobs from the MHS_TM hold category.
MHS_USER DISPLAY	Displays MHS_USER information for the job specified or for all jobs held by MHS for the MHS_USER category.
MHS_USER RELEASE	Removes jobs from the MHS_USER hold category.
MHS_VOL DISPLAY	Displays MHS_VOL information for the job specified or for all jobs delayed by MHS for the MHS_VOL category.
MHS_VOL RELEASE	Removes jobs from the MHS_VOL delay category.

Language Services

Chapter 13. Job Action Language (JAL) Function

This chapter describes how to include JAL in your ThruPut Manager configuration. To find information describing how to write the JAL to implement your installation's batch processing rules, see the ***DAL/JAL User Guide*** *and the* ***JAL Reference Guide****.*

Description

The Job Action Language is the heart of ThruPut Manager. It is the means for your installation to apply system management to your batch workload. JAL provides you with a set of comprehensive variables and built-in functions that allow you to accurately categorize each job. In this chapter, we are only concerned with the *actions* that can be requested with JAL for each job processed by ThruPut Manager. For a description of Job Action Language capabilities, refer to the ***DAL/JAL User Guide*** and the ***JAL Reference Guide***.

JAL and Optional Components

JAL for the ThruPut Manager Base Product supports Job Classing. Additionally, there are extensions to support other components. As part of the Base Product, there are JAL extensions to support:

- Job Setup Services (JSS)

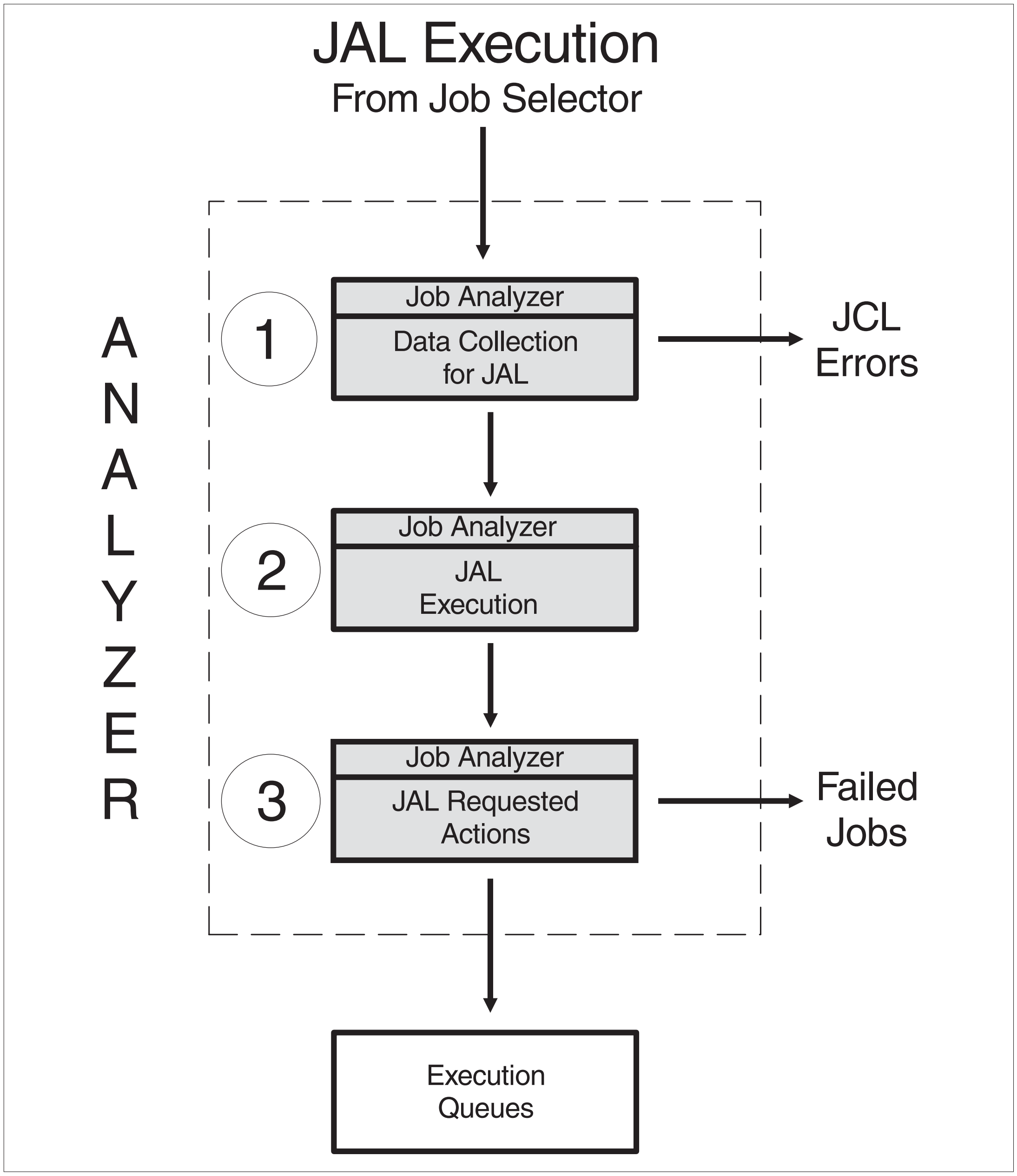
Additionally, your installation might add one or more of the ThruPut Manager optional components:

- Dataset Contention Services (DCS)
- Job Binding Services (JBS)

JAL includes extensions to support each of these components. Details describing these extensions are provided later in this chapter.

JAL Execution—Control Flow

An overview of the control flow for the execution of JAL is included here to illustrate where JAL fits in the overall architecture of ThruPut Manager:



JAL Control Flow

JAL Actions

As the diagram shows, JAL can request that ThruPut Manager take actions for each job processed. The following sections provide a summary of the possible actions.

Base Product (Job Classing)

With the ThruPut Manager Base Product, you can perform job verification, and depending on the outcome, request ThruPut Manager to:

- Insert warnings or information messages in JOBLOG/SYSMSGS and let the job go to execution.
- Place the job in JES2 HOLD status.
- Fail the job.
- Assign the execution class.
- Assign a class for Deferred Processing Services.
- Assign the execution priority.
- Set a CPU time limit for the job.
- Assign a performance group.
- Produce a simple list of mountable volumes and send it directly to an output device.

Job Setup Services Component Extensions.

JAL is extended to support the management of mountable volume requests. You can:

- Request a Volume List for a job that requires mountable volumes, causing the job to be placed in VOLUME hold and its Volume List to be queued for printing. A job in VOLUME hold is released automatically when media-handlers signal that its volumes are available.
- Accumulate mountable volumes for several jobs so their volumes can all be requested together. The resulting Consolidated Volume List can then be “spooled” and printed.
- Initiate asynchronous DFHSM requests for jobs with migrated datasets, and apply an HSM hold to allow time for the dataset to be recalled.
- Do further analysis of jobs based on automated or manual cartridges.

TM/Dataset Contention Services Component Extensions

TM/DCS Only

There are a number of significant extensions to JAL to support the TM/DCS Component. They allow you to:

- Set the service level that TM/DCS gives a job.
- Determine the characteristics of a dataset and the DD statement that references it, so that you can use this information to decide how TM/DCS should manage contention for the dataset.
- Repossess a required dataset from TSO users who have the dataset opened for input.
- Request NAGs and ALERTs to help resolve dataset contention problems.
- Request dataset contention recording for a job.

TM/Job Binding Services Component Extensions

TM/JBS Only

There are a number of significant extensions to JAL to support the TM/JBS Component. They allow you to:

- Determine whether or not a job has JECL statements, the types of the statements, and naming conventions for Binding Agents, Limiting Agents, and Batch Names.
- Add Binding statements to a job.
- Add Limiting Agents to a job.
- Determine whether or not you have “inserted” any BIND statements.
- Relate the job to a Limiting Agent.
- Determine whether or not you have reached the total number of BIND statements that you can insert (up to 24) or the number of Limiting Agents you can associate with a job (up to 24).
- Assign Batch Names to jobs so they can be grouped to define the scope of Before & After actions.

Implementation Summary

You must convert JAL source statements into a format that ThruPut Manager can use. This is the purpose of the Language Processor. Your JAL source statements are the input to the Language Processor, and the output is JAL internal text, suitable for use by ThruPut Manager. For details about using the Language Processor, refer to the ***DAL/JAL User Guide***.

The sequence required to implement JAL is:

1. Write JAL statements defining your installation's rules.
2. Run the Language Processor to verify your JAL source statements, and convert them into the internal text required by ThruPut Manager.
3. Add the statement JAL LOAD to the TMSS initialization statements. This indicates that there are installation rules and where they can be found (JAL internal text).

☞ ***To load a new JAL does not require that you start TMSS again. The JAL REFRESH command lets you refresh JAL dynamically.***

Dependencies

Internal JAL text must be provided to TMSS at startup time. The JAL LOAD initialization statement indicates where to find the internal text created by the Language Processor. The Language Processor is a separate component of ThruPut Manager, and is described in detail in the ***DAL/JAL User Guide***.

TMSS Initialization Statements

There are two TMSS initialization statements for this function:

```
JAL LOAD
JAL TRACE
```

They are documented in "Chapter 2. TMSS Initialization Statements."

Facilities Summary

JAL Initialization Statements		
Statement	**Description**	**Page**
JAL LOAD	Specifies the dataset name and member name (if applicable) where the JAL internal text is located.	57
JAL TRACE	Allows you to request JAL trace at TMSS start up time. You can trace specific jobs or all jobs.	58

JAL Operator Commands	
Command	**Description**
JAL DISPLAY	Displays the active JAL and its identifier.
JAL REFRESH	Allows you to activate a new JAL dynamically.
JAL TRACE	Allows you to activate tracing for all jobs, or selected jobs.

JAL JECL Statement		
Statement	**Description**	**Page**
/*JAL TRACE	Activates JAL tracing for this job.	

PRE–JAL EXITS		
Exit #	**Exit Name**	**Description**
Exit 1	JCL Job Statement	This exit receives control when the Job Analyzer is about to process a JCL JOB statement.
Exit 2	JCL EXEC Statement	This exit receives control for every EXEC statement in a job.
Exit 3	JCL DD Statement	This exit receives control for each DD statement in a job.
Exit 4	UNIT NAME/DEVICE TYPE	This exit receives control every time the Job Analyzer encounters a new unit name/device type that influences its resource analysis.
Exit 5	Volume Status	This exit receives control every time the Job Analyzer encounters a *unique volume serial number.*

POST-JAL EXITS		
Exit #	**Exit Name**	**Description**
Exit 6	JAL Action Modification	This exit receives control after the ThruPut Manager Job Analyzer has classified a job using the JAL supplied by the installation.
Exit 7	Job Information Collection	This exit receives control when ThruPut Manager is ready to save all the information collected about a job.

Chapter 14. Detail Action Language (DAL) Function

*This chapter describes how to include DAL in your ThruPut Manager configuration. To find information describing how to write the DAL to implement your installation's batch processing rules, see the **DAL/JAL User Guide** and the **DAL Reference Guide**.*

Description

Detail Action Language, or DAL, is a ThruPut Manager Action Language that allows you to exercise control over your batch workload at a *detailed* level–the dataset/DD statement level. This contrasts with JAL, which provides control only at the job level.

Because it operates at a more detailed level, DAL gives you the opportunity to perform a more specific analysis:

- DAL works in conjunction with JAL and is invoked during the Job Analysis process every time a DD statement is encountered.
- JAL, on the other hand, is executed only once per job, after all JCL statements have been processed by the Job Analyzer.

DAL allows you examine each DD statement in a job. You can verify statement parameters, issue messages, and if necessary, fail the job. If desired, you can also set User Logic Variables to communicate the results of your examinations to JAL or ***subsequent*** invocations of DAL.

Types of DAL

Some of the actions you might want to take apply only to certain types of DD statements. For example, if you want to control tape cartridge automation, there is no point in examining a DD statement that describes a DASD dataset. Invoking your tape automation logic just for tape DD statements is not only more efficient, it also allows you to change or replace it without affecting processing rules intended for other purposes.

There are therefore different types of DAL, each associated with a particular ThruPut Manager component. The choice of which one to use depends on what you plan to do and, if necessary, which optional ThruPut Manager component you are using.

- ***TM DAL***, associated with the base level of ThruPut Manager, lets you examine *every DD statement in a job*. You can verify statement parameters, issue messages, and if necessary, fail jobs.
- ***DCS DAL***, associated with TM/DCS, gives you control over dataset contention and is invoked *only for permanent datasets*. You can take into account the dataset, the requesting job, and the current holder of the dataset. DCS DAL can issue Alert messages to the owners of jobs whose datasets have encountered contention, issue Nag messages to TSO users holding required datasets, and if necessary, alter the disposition or even repossess required datasets.
- ***SOS DAL***, associated with TM/SOS, provides the ability to manage the complex relationships of DD statements specifying the SYSOUT keyword (SYSOUT DD) with OUTPUT JCL statements, /*OUTPUT JECL statements, and the system defaults for SYSOUT datasets. You can examine the characteristics of a SYSOUT dataset after the effects of JCL OUTPUT, JECL /*OUTPUT, and system defaults have been applied, then alter characteristics based on your analysis.

In this chapter, we are concerned only with the *actions* that can be requested with DAL. For a description of the Detail Action Language capabilities, refer to the ***DAL/JAL User Guide*** and the ***DAL Reference Guide***.

DAL Execution–Control Flow

When you include DAL of any type, upon encountering a DD statement the Job Analyzer follows the control flow explained below.

In this extended environment, every time a DD statement is processed, the following additional actions take place:

1. All the associated Descriptors are built from the JOB statement and the EXEC statement previously analyzed.
2. The DD User Exit is taken (the same as before).
3. A Job Profile is built for DAL.
4. The environment is prepared for DAL.
5. DAL is executed.
6. Control returns to continue JCL analysis, unless you decide to FLUSH the job during the execution of DAL.

When DAL or JAL is invoked, it is executed in one pass. You always "move forward" in your processing logic, so it is not possible to accidentally create programming loops. Remember, there is a significant difference between JAL and DAL:

- JAL is invoked only once per job.
- DAL is invoked for every DD statement in the JCL, as follows:
  - If present, TM DAL is invoked first, and is called for ***every DD statement in the job***.
  - Next, if present, DCS DAL is invoked ***every time a DD statement for a permanent dataset is processed***. It specifies how you want TM/DCS to handle dataset contention problems.
  - Finally, if present, SOS DAL is invoked ***at least once for every DD statement specifying the SYSOUT keyword. If the SYSOUT DD is affected by more than one OUTPUT JCL statement, SOS DAL is invoked once for each unique pairing.*** It allows you to examine and alter characteristics of the SYSOUT dataset.

DAL and JAL—Rule Precedence

ThruPut Manager gives you facilities in JAL to control the job. ***These rules are at the job level,*** that is, whatever instructions you give to ThruPut Manager apply to the entire job.

If, on the other hand, you just want give instructions ***for only one particular DD statement in the job***, then you need the DAL facilities.

DAL allows a "finer degree of granularity" than JAL so that you have more control over the job. For example, you can provide rule for situations covering relationships between a particular dataset and a job, or a particular cartridge and a job.

There are a number of requests that you can make either in DAL or JAL. Since DAL is a more precise instrument than JAL, DAL rules have precedence over rules specified in JAL. In other words, JAL rules apply only to circumstances that have not been handled in DAL.

Implementation Summary

You must convert DAL source statements into a format that ThruPut Manager can use. This is the purpose of the Language Processor. Your DAL source statements are the input to the Language Processor, and the output is DAL internal text, suitable for use by ThruPut Manager. For details about using the Language Processor, refer to the ***DAL/JAL User Guide***.

The sequence required to implement DAL is:

1. Write DAL statements defining your installation's rules.
2. Run the Language Processor to verify your DAL source statements, and convert them into the internal text required by ThruPut Manager.
3. Add the statement DAL LOAD to the TMSS initialization statements. This indicates that there are installation rules and where they can be found (DAL internal text).

   ☞ ***To load a new DAL does not require that you start TMSS again. The DAL REFRESH command lets you refresh DAL dynamically.***

Dependencies

If used, internal DAL text must be provided to TMSS at startup time. The DAL LOAD initialization statement indicates where to find the internal text created by the Language Processor. The Language Processor is a separate component of ThruPut Manager, and is described in detail in the ***DAL/JAL User Guide***.

TMSS Initialization Statements

There are two TMSS initialization statements for this function:

```
DAL LOAD
DAL TRACE
```

They are documented in "Chapter 2. TMSS Initialization Statements."

Facilities Summary

DAL Initialization Statements		
Statement	**Description**	**Page**
DAL LOAD	Specifies the dataset name and member name (if applicable) where the DAL internal text is located.	43
DAL TRACE	Allows you to request DAL trace at TMSS start up time. You can trace specific jobs or all jobs.	44

DAL Operator Commands	
Command	**Description**
DAL DISPLAY	Displays the active DAL and its identifier.
DAL REFRESH	Allows you to activate a new DAL dynamically.
DAL TRACE	Allows you to activate tracing for all jobs, or selected jobs.

DAL JECL Statement		
Statement	**Description**	**Page**
/*DAL TRACE	Activates DAL tracing for this job.	

Chapter 15. Table (TBL) Function

This chapter describes the DAL/JAL table management facilities.

Description

In some cases, your installation might have a need to ensure that a Descriptor (RACF id, for example) matches a particular value from a number of values that are not readily suited to pattern matching. To accommodate this, DAL and JAL support the concept of an *external table*.

External tables can contain all legal values that need to be checked, reducing DAL or JAL coding to the single test of whether the Descriptor value is contained in the table or not. Because the tables are external, management is simplified. Changes to the table do not require alterations to the DAL or JAL code, eliminating the possibility that logic errors might be introduced during updates. Operator commands provide the ability to display table status, refresh tables at any time, or if necessary, to disable an active table.

To support table management, the following facilities are included in DAL and JAL:

- A logic construct that allows you to specify a *table* as the argument to ***any*** pattern matching function.
- A logic variable that allows you to determine whether or not a given table has been loaded.

To the ThruPut Manager component that does the initial table loading, a table is a sequential file or a PDS member containing:

- Table entries, which are character strings, or patterns to match character strings or dataset names, as described in the ***DAL/JAL User Guide***.
- Subtable identification statements, which simplify management and editing of tables by letting you segment a table into subtables that can be processed separately or together, depending on your DAL or JAL.

A full description of the facilities to support DAL/JAL external tables is provided in the publication ***DAL/JAL User Guide***. The commands to refresh, display, enable, and disable tables are described in the ***Operating Guide***.

Implementation Summary

No special action is required to install or activate DAL/JAL table support. To implement a specific table:

- Create a PDS member or sequential dataset containing the character strings or patterns that are valid. The syntax and rules for building tables are documented in the ***DAL/JAL User Guide***.
- Ensure that the table is loaded. Normally, you would add a TBL LOAD statement for the PDS member or sequential dataset to the TMSS initialization statements. Note that the TMSS initialization statement is not mandatory, however, since tables can be loaded dynamically by using operator commands.
- Add code to your DAL or JAL to test for the presence of the table, and to use the table if present.

Dependencies

There are no other dependencies for DAL/JAL table support.

TMSS Initialization Statement

The TMSS initialization statement TBL LOAD is used to load tables when TMSS is first started.

```
TBL LOAD n dsname[(member-name)]
```

This statement loads the table or tables contained in a sequential dataset or PDS member. The TBL LOAD initialization statement is documented fully in “Chapter 2. TMSS Initialization Statements.”

Facilities Summary

TBL Initialization Statement		
Statement	**Description**	**Page**
TBL LOAD	Specifies the dataset name and member name (if applicable) where the external Table text is to be loaded from.	65

TBL Operator Commands	
Command	**Description**
TBL DISABLE	Allows you to disable a previously loaded table or subtable.
TBL DISPLAY	Displays information about the status of a table.
TBL ENABLE	Allows you to enable a previously loaded but subsequently disabled table or subtable.
TBL REFRESH	Loads a new copy of a previously loaded table.

The Job Classing Component

Chapter 16. Job Selection Function

This chapter describes the Job Selection Function process of ThruPut Manager. It also documents the necessary steps to customize the process to reflect your installation needs.

The Job Selection Function

The Job Selection Function allows you to define to ThruPut Manager how to handle jobs based on their submission class or type.

Classes for the Job Selector

Two “types” of classes have particular importance for the Job Selector:

1. The Job Analysis Class.
2. Select Classes.

The first one represents a unique class that is allocated ***for ThruPut Manager usage only***. The default is 9, but it is installation definable.

The Select Classes represent the list of classes that your installation has chosen to be processed by ThruPut Manager. The complement of Select classes is the Exempt classes. That is, a class is either in the Select group or in the Exempt group. Unless you change the default, ***all classes*** are processed by ThruPut Manager.

The Job Analysis Class and the Select classes (or Exempt if you prefer) are defined with the TMPARM JES2 statement.

The ability to exempt classes from ThruPut Manager processing has been provided mainly for the initial product implementation phase. This allows you to start with a few classes to ensure things are working to your satisfaction. Later, you can add classes in increments until you have defined all the rules necessary for ThruPut Manager to manage your batch workload.

☞ ***You can use the TM CLASS operator command to display or alter the classes to be selected for ThruPut Manager processing.***

There is one more group of submission classes in a ThruPut Manager environment. This group includes the Deferred classes, and is explained in “Chapter 18. Deferred Processing Services (DPS) Function.”

Job Selection Considerations

Jobs that are submitted to one of the classes under the control of ThruPut Manager are processed by the Job Analyzer. Conversely, jobs that belong to Exempt classes do not go through the Job Analysis process. As a result, these jobs do not benefit from the JCL verification services provided by ThruPut Manager.

☞ ***If you $TJ a job that is queued in the Job Analysis Class to another class, you are bypassing ThruPut Manager processing. This might result in a job failure if the job contains TM JECL statements.***

There are other types of jobs that require special consideration:

- Jobs submitted with TYPRUN=HOLD are held before the CONVERTOR phase, therefore they do not receive ThruPut Manager processing until they are released. You can request that jobs of this type be analyzed as soon as they are submitted. This is done with the TMPARM JES2 statement.
- Jobs submitted with TYPRUN=SCAN are terminated by the system after the CONVERTOR phase. As a result, their JCL is not checked by ThruPut Manager. You can request that jobs of this type receive full JCL verification. This is done with the TMPARM JES2 statement.
- Jobs that include a /*SETUP statement are held by JES2 after conversion. As a result, they are not processed by ThruPut Manager until they are released. You can request that jobs of this type be processed by ThruPut Manager as soon as they are submitted. This is done with the TMPARM JES2 statement.

EXEMPT Class Jobs and JECL Statements

For some functions of ThruPut Manager, users can include JECL JES2 control statements in their JCL. The statements are collected during the JES2 READER phase to be processed by the Job Analyzer. As indicated before, jobs that are submitted in Exempted classes are not normally processed by ThruPut Manager.

☞ ***If jobs submitted in Exempted classes include ThruPut Manager JECL statements, then they are processed by the Job Analyzer unless the JECLEXEMPT keyword has been included on the TMPARM JES2 initialization statement.***

At JAL time you can test a logic variable ($EXEMPT) that indicates whether or not the job belongs to an Exempted class. This allows you to detect jobs that belong to an Exempted class but have JECL statements. You can then take appropriate actions.

Implementation Summary

The Job Selection Function becomes an integral part of your system when you install and activate the Job Selector component.

When you follow the steps required to activate the Job Selector, you do the following:

- Decide which class to use for the Job Analysis Class.
- Decide which classes are to be Selected (or Exempted).
- Put the procedures in place to assign the Job Analysis Class to an initiator.

After that, the Job Selection Function is in place and ready to process jobs.

Dependencies

There are no other dependencies for this function.

TMPARM Keywords for this Function

The following keywords apply to this function:

```
ANALYSIS
DEFER
EXEMPT
JECLEXEMPT
SELECT
SETUP
TYPSCAN
TYPHOLD
```

For a detailed syntax description, refer to the TMPARM statement description in the ***Installation Guide***

TMSS Initialization Statements

There are no TMSS initialization statements for this function.

Facilities Summary

TMPARM Statement Refer to *Installation Guide*		
Keywords	**Description**	**Chapter**
ANALYSIS	Defines the Job Analysis Class.	3
DEFER	Allows you to define Deferred classes.	3
EXEMPT	Allows you to define the classes to be exempted for ThruPut Manager processing.	3
SELECT	Allows you to define the classes to be selected for ThruPut Manager processing.	3
SETUP	Defines the selection for jobs submitted with a /*SETUP JES2 control statement.	3
TYPSCAN	Defines the selection for jobs submitted with TYPRUN=SCAN.	3
TYPHOLD	Defines the selection for jobs submitted with TYPRUN=HOLD.	3

Job Selection Function Operator Commands	
Command	**Description**
TM CLASS	Allows you to display or alter the classes to be selected for ThruPut Manager processing.

Chapter 17. Job Display Services (JOB) Function

This chapter describes the Job Display Services and the different types of information that can be displayed with the JOB DISPLAY command.

Description

The Job Services (JOB) Function provides the following

- A display of certain job-related information.
- The ability for your installation to format up to ten lines of information about a job and make it available for display by the JOB DISPLAY operator command.

Job-related Data Display

Certain job-related information is always displayed. This information is designed to answer questions such as:

- What class and priority did the job receive?
- Is the job held by JES2, and if so, was the job held because of a HOLD ALL command or a specific HOLD for the job?
- Is the job held by ThruPut Manager, and if so, which ThruPut Manager applications caused the job to be held?
- Does the job require any UNAVAILABLE volumes, and if so, which ones?
- What DASD volumes does the job use?
- What tape volumes does the job require?

Installation Data Display

The most significant aspect of Job Display Services is that it provides you with the ability to format and make available for display whatever information is important to your installation.

All the information collected by the Job Analyzer is available for formatting. Using JAL facilities, you can control exactly what information is available for display and how it is formatted. The information remains available until the job is purged from the system.

Up to ten lines of installation data can be defined. Each line can be designated for either ***automatic*** or ***requested*** display:

- ***Automatic*** display means that the line is displayed automatically when the JOB DISPLAY command is issued.
- ***Requested*** display means that the line is displayed only if the JOB DISPLAY command includes the ONREQUEST keyword.

The JOB DISPLAY Command

This command displays job-related information and any installation information that has been formatted in JAL for ***automatic*** display. It also provides various operands to support inquiries about specific information. These operands can be combined on the same command to obtain more data.

The following operands can be specified:

DASD | D

To display all the DASD volumes required by the job.

ONREQUEST | ONREQ

To display installation data that was formatted by JAL for display only when requested.

TAPES | T

To display all the tape volumes required by the job.

Examples

The following examples show both the output from the JOB DISPLAY command and the method used to prepare installation data for display.

Job-related Data

For simplification, the first example shows the output for a job for which there is no installation data. To display the job-related information for job number 1234:

```
/JOB DISPLAY 1234
```

A typical response to this command is:

```
DTM0220I JOB DISPLAY
1234 MYJOB  AWAITING EXECUTION  E  PRIO 9  ANY
JES2-HOLD(ALL)  TM-HOLD
MANAGED:  JSS    HOLD
```

The first line of the display is always present, and includes the job name, job number, JES2 queue, class, priority, and system affinity.

If the job is held, a line is included showing whether the job is held by JES2 or ThruPut Manager (or both), and if by JES2, whether the hold resulted from a HOLD ALL command or a specific hold for the job.

If the job is held by a ThruPut Manager application, additional lines are included to show which applications are managing the job.

This portion of the output is always the same, regardless of any operands specified or whether or not there is installation data for the job.

Volume Information

Again for simplification, this example assumes that there is no installation data for the job. To display the resident DASD volumes required by a job:

```
/JOB DISPLAY 1234 DASD
```

This command produces the job-related data display described above, with additional lines in the following format:

```
UNAVAILABLE DASD VOLUMES
  RES202
RESIDENT DASD VOLUMES
  RES177     RES092
```

Any volumes that ThruPut Manager has determined to be UNAVAILABLE are listed first, because these references normally indicate a problem exists with the job. For a discussion of how ThruPut Manager determines which volumes are UNAVAILABLE, refer to “Chapter 21. Volume Status (VOL) Function.”

To display the mountable volumes required by a job:

```
/JOB DISPLAY 1234 TAPES
```

This command produces the job-related data display described previously, with additional lines in the following format:

```
TAPE VOLUMES
  TC2109     TC3498     TC7116     TC5981
```

You can specify both the DASD and TAPES keywords on the same command. If you do this, the DASD display is produced first, regardless of the order in which you specify the keywords.

Installation Data

In the previous examples, we assumed that there was no installation data so that the explanations could be kept simple. One reason for this is that if there is installation data, it is displayed immediately after the job-related data but *before* any volume information. We will use our first sample command again:

```
/JOB DISPLAY 1234
```

This time, we will assume that some installation data has been formatted in JAL and associated with this job. A typical response might be:

```
DTM0220I JOB DISPLAY
1234 MYJOB  AWAITING EXECUTION  E  PRIO 9  ANY
JES2-HOLD(ALL)  TM-HOLD
MANAGED:  JSS    HOLD
TM INSTALLATION INFORMATION
Submission Class     A     Assigned Class      E
Submission Priority  7     Assigned Priority   9
CPU Time:      0:10 MINS
```

Note that under the heading TM INSTALLATION INFORMATION, the response includes any lines formatted for ***automatic*** display. These lines are prepared as follows.

Using JOB_DATADEF to Format Installation Data

The JOB_DATADEF JAL definition statement allows you to include any information collected during Job Analysis, as well as hard-coded text, and to align the data as needed. For the example above, the JAL statements would be:

```
CLASS_LINE  JOB_DATADEF  ('Submission Class',(COL,22,L),$INCLASS    +
                          (COL,28,L),                               +
                          'Assigned Class',(COL,48,L),$JXCLASS)
PRIO_LINE   JOB_DATADEF  ('Submission Priority',(COL,22,L),$INPRIO  +
                          (COL,28,L),                               +
                          'Assigned Priority',(COL,48,L),$JXPRIORITY)
CPU_LINE    JOB_DATADEF  ('CPU Time:',(COL,20,R),$JOBCPU,' MINS')
```

Using JOB STORE to Make Data Available for Display

In your JAL logic, you determine whether you want to associate each line with the job under analysis. In this example, we assume that you want to display all of the defined lines for our sample job. Once these decisions have been made, you need to associate the lines with the job. Use the JOB STORE action statement:

```
JOB STORE OPER_DATA(LINE1(CLASS_LINE),LINE2(PRIO_LINE),LINE4(CPU_LINE))
```

This stores the three lines with the other information for the job collected during Job Analysis. They are displayed *automatically* when you issue a JOB DISPLAY command.

Data Displayed by Request Only

Note that in our example above, the third line (CPU_LINE) is specified as LINE4. This shows that lines need not be defined or stored in sequence. In our example, LINE3 has been omitted because we will use it to illustrate another form of installation information: ***requested*** data.

The definition of a line of data that is only available upon request is identical to other lines:

```
RACF_LINE   JOB_DATADEF  ('RACF Group:',(COL,16,L),$RACFG,  +
                          (COL,28,L),                       +
                          'RACF Userid:,(COL,44,L),$RACFU)
```

Making the line available for display *only upon request* is done when the line is associated with a job by the JOB STORE action statement:

```
JOB STORE OPER_DATA(LINE3(RACF_LINE)) ON_REQUEST
```

To display installation data lines that have been associated with the job for ***requested*** display, the command is:

```
/JOB DISPLAY 1234 ONREQUEST
```

This command produces a display virtually identical to the one described previously, except that it includes the installation lines that were intended for display upon request only:

```
TM INSTALLATION INFORMATION
Submission Class     A     Assigned Class     E
Submission Priority  7     Assigned Priority  9
RACF Group:    DEV         RACF Userid:    Z99TIM
CPU Time:      0:10 MINS
```

Note that the line formatted for ***requested*** display was interspersed with the lines formatted for ***automatic*** display.

For further information about the use of JOB_DATADEF and JOB STORE for the display of installation data, refer to the ***DAL/JAL User Guide*** and the ***JAL Reference Guide***.

Implementation Summary

The steps needed to implement the display of installation data are:

- Decide what data is useful to display and design standard display formats.
- Code JAL to define the information you want displayed and to associate that data with a particular job. Installation data is defined using the JOB_DATADEF definition statement.
- Code JAL logic to determine whether the data defined by JOB_DATADEF is to be made available for display for the job under analysis.
- If the data is to be available for display, use the JOB STORE action statement to associate the line with the job. JOB STORE allows you to specify whether the line is available for ***automatic*** display or only for ***requested*** display.
- Document the displays and make the documentation available to those who will be using the JOB DISPLAY command.

Dependencies

There are no dependencies for job-related data display, except that the job must have been processed by ThruPut Manager (non-Exempt). For installation data display, JAL must be coded to format the data lines and make them available for display.

TMSS Initialization Statements

There are no initialization statements required to activate this function.

Facilities Summary

JOB Operator Commands	
Command	**Description**
JOB DISPLAY	Allows you to display information about a particular job.

JOB EXITS		
Exit #	**Exit Name**	**Description**
Exit 13	Display Job	This exit receives control when the JOB Function is processing a DISPLAY command.

Chapter 18. Deferred Processing Services (DPS) Function

This chapter describes Deferred Processing, and how to implement and use it.

Description

Deferred Processing is a convenient way to group together jobs which are to be given "delayed services" such as overnight processing. This technique eliminates complex instructions to operators concerning when to hold or release jobs, and also ensures that Volume Lists contain the most current information available.

Suppose that you want to implement an overnight service class. Jobs submitted for overnight processing are to be processed after 10:00 PM, so they can be ready for the next business day. You have to do the following:

- Set criteria to decide which jobs are to receive deferred processing. In this example users request the service by submitting jobs in class 'D'.
- Choose a class to be used as ***the class to "park" the deferred jobs***. This is called a Deferred Class. The jobs are "parked" in the Deferred Class between the time they are submitted and the time you begin processing. The Deferred Class, like the Job Analysis class, is dedicated exclusively to ThruPut Manager. In this example, let's say it is '8'.

In your JAL logic, the first thing you should do is to detect jobs that are submitted for deferred processing. At the time of submission you only want JCL verification. You do not want Volume Lists or any other ThruPut Manager processing, since the job is not eligible for execution yet. Your JAL might look like this:

```
...
EVALUATE   DEFER  ($INCLASS(D) & $CURCLASS(*))
...
IF (DEFER)       /*IS THIS FIRST PASS FOR OVERNITES?*/
  SET CLASS(8)   /*PARK JOB IN OVERNITE CLASS*/
  EXIT REQUEUE   /*EXIT IMMEDIATELY*/
ENDIF
...
```

The Job Property 'DEFER' represents jobs that have been submitted for overnight processing (the submission class is 'D') but they are being processed by

ThruPut Manager for the first time (the current class '*' represents the primary Job Analyzer class).

Jobs submitted for "overnight" service can be "parked" in the "overnight" class immediately (in your JAL logic). In this way, because they have been processed by the Job Analyzer, ***their JCL has been verified to ensure there are no errors.*** Since you have requeued them in the overnight class they can be "reprocessed" by ThruPut Manager at the appropriate time.

At 10:00 PM, your operations group does the following:

- They assign the Deferred Class '8' to an initiator.

This way all the jobs that were "parked" in class '8' for deferred processing are reprocessed by ThruPut Manager. In your JAL (shown again below), the Job Property 'DEFER' is not true:

```
...
EVALUATE   DEFER  ($INCLASS(D) & $CURCLASS(*))
...
IF (DEFER)       /*IS THIS FIRST PASS FOR OVERNITES?*/
   SET CLASS(8)  /*ASSIGN PARKING CLASS*/
   EXIT REQUEUE  /*REQUEUE THEM IMMEDIATELY*/
ENDIF
...
```

The current class of the job is the Deferred Class '8', hence the EVALUATE statement returns a value of FALSE. This time the job can be fully processed by ThruPut Manager.

To stop accepting overnight jobs to be processed that night, your operations group does the following:

- They remove the Deferred Class '8' from the Job Analyzer initiator.

This approach works well but it can be improved. For jobs submitted for overnight processing *that arrive in the system after 10:00 PM, there is no point in giving them two ThruPut Manager passes.* They are eligible for processing as soon as they arrive. The external EVENT switch facility has been designed to help and complement this aspect of Deferred Processing. This is best illustrated by a simple example. The EVENT switch 'OVERNITE' (or any other name you find appropriate) is used to control the one-pass two-passes ThruPut Manager processing.

Your JAL might look like this:

```
...
EVENT   OVERNITE(OFF)   /*INITIAL STATE FOR EVENT*/
...
EVALUATE DEFER ($INCLASS(D) & $CURCLASS(*)) & OVERNITE
...
IF (DEFER)
  SET CLASS(8)    /*ASSIGN PARKING CLASS*/
  EXIT REQUEUE    /*REQUEUE THEM IMMEDIATELY*/
ENDIF
...
```

For the overnight jobs, the need for the second pass is established by the status of the EVENT OVERNITE switch. Now the procedure for initiating overnight processing is as follows:

At 10:00 PM, your operations group does the following:

- They assign the Deferred Class ‘8’ to the Job Analyzer initiator.
- They set the OVERNITE EVENT switch to ‘ON’ with the ThruPut Manager command ‘JAL EVENT ON OVERNITE’.

Because the OVERNITE switch is ‘ON’ the Job Property ‘DEFER’ returns a value of ‘FALSE’ so the overnight jobs are not placed in the “parking” class.

In situations where overnight processing is available, your installation may have a cut off point for submission. In this example, let us say it is 3:00 AM. Any job submitted for overnight processing after 3:00 AM is again “parked” until the following night.

To stop accepting work to be processed that night, at 3:00 AM your operations group does the following:

- They set the OVERNITE EVENT switch to ‘OFF’ with the ThruPut Manager command ‘JAL EVENT OFF OVERNITE’.
- Then, they remove the Deferred Class ‘8’ from the Job Analyzer initiator.

Deferred Processing Analyzers and Workload Manager

Once a $ACTIVATE has been done, there are multiple operating strategies available.

Using JES2 Initiators for Deferred Processing Classes

You can choose to continue just as you did before the $ACTIVATE. This is the simplest choice because there are no changes to the way things worked previously.

Using WLM Initiators for Deferred Processing Classes

If you allow WLM to manage the initiators for your Deferred Processing class, there are several considerations:

- You can assign the same service class you assigned for Job Analysis, or define a separate service class for the Deferred classes. Because deferred jobs by definition do not need immediate service, a separate service class can have less aggressive goals but still should deliver fast service to short-running transactions. The service class routine should ensure that jobs in the Deferred Processing job class are assigned the correct service class.
- ***Do not use this service class for purposes other than job analysis.***
- WLM controls the number of Deferred Processing initiators only at the JESplex level. You can control the total number of Job Analyzers by setting the XEQCOUNT for the Deferred class, but control at the system level is not available.
- You can also control the availability of Deferred Processing initiators by defining a scheduling environment for the Deferred Processing class that is not always available.
- Because Deferred Processing accumulates jobs, consider the potential WLM reaction when the Deferred jobs are eligible for processing. For example, even if you choose to control the Deferred analysis by the availability of a scheduling environment, you might want to set the XEQCOUNT for the Deferred class to a value that ensures that WLM does not start too many initiators. You might also consider a service class that ignores queue time.

Implementation Summary

The steps for implementing Deferred Processing are:

- Specify at least one Deferred Job Analysis Class for the DEFER keyword of the ThruPut Manager JES2 initialization statement TMPARM. You need one Deferred Class per Deferred Service you want to implement. For example, if you want to provide overnight services and weekend services you need two Deferred Classes. See ***Installation Guide for TMPARM definitions.***

- Code JAL to select jobs which should be deferred, and queue these jobs to a Deferred Job Analysis Class.
- When jobs which have been deferred are to be processed, add the Deferred Job Analysis Class to the initiator assigned to the Job Analyzer.
- When accepting Deferred Processing is to stop, remove the Deferred Class from the initiator assigned to the Job Analyzer.
- Determine if you want to control the flow of the Job Analysis for the Deferred Service with an EVENT switch.
- If EVENT is to be used, put the appropriate operating procedure in place to turn the chosen EVENT "on" and "off" at the appropriate times.
- Add logic to your JAL to test for the status of the EVENT switch and take actions accordingly.

Dependencies

You must ensure that procedures are in place to add the deferred class, or classes, to the Analyzer initiator at the appropriate time of the day.

Depending on your implementation, you could have dependencies on EVENTS.

TMPARM Keywords for this Function

You have to specify at least one DEFER class in your JES2 initialization statement TMPARM. See ***Installation Guide***

TMSS Initialization Statements

Deferred Processing is not dependent on any TMSS initialization statements.

Facilities Summary

TMPARM Statement Refer to *Installation Guide*		
Keywords	**Description**	**Chapter**
DEFER	Defines the classes to be used for deferred processing.	3

DPS Operator Commands	
Command	**Description**
JAL EVENT	Allows you to display and alter the status of an EVENT switch.

Chapter 19. Job Validation Services (JVS)

This chapter explains how JCL checking at submission time can be extended to jobs that are normally held before Job Analysis.

Introduction

In a system with ThruPut Manager, comprehensive JCL checking is provided as soon as the job is submitted. The early JCL checking facility applies to all jobs processed by ThruPut Manager. If an error is found, an appropriate message is provided, and the job is failed as soon as it is submitted. With this facility there are no costly reruns and no resources wasted.

There are certain type of jobs that do not receive the benefits from the early JCL verification, unless you instruct ThruPut Manager to process them too. There are three distinct groups:

- Jobs submitted with TYPRUN=SCAN.
    - SCAN jobs receive a cursory JCL checking, where gross errors can go undetected. You can choose to give these jobs full ThruPut Manager JCL checking.
- Jobs submitted with TYPRUN=HOLD.
    - Jobs that are submitted with HOLD do not receive full JCL checking until execution time. With ThruPut Manager, you can choose to give these jobs full JCL checking at submission time.
- Jobs submitted with a /*SETUP JES2 control statement.
    - Jobs submitted with a /*SETUP statement are normally held before Job Analysis. You can choose to give jobs of this type ThruPut Manager processing as soon as they enter the system.

JCL Validation Service

The installation can now offer a JCL Validation Service to their users. Jobs that are submitted with TYPRUN=SCAN can receive the ThruPut Manager's comprehensive JCL verification. If any error is encountered, informative messages are included with the JCL listing. This is a ThruPut Manager option that your installation can activate.

Implementation Summary

Keywords are provided with the JES2 TMPARM statement to define what you want ThruPut Manager to do with TYPRUN=SCAN, TYPRUN=HOLD and /*SETUP jobs.

SETUP

This keyword is used to indicate to ThruPut Manager that jobs using the /*SETUP statement should not be held. ***Note that the standard /*SETUP message is sent to the console.*** If you want the jobs to be held after Job Analysis, you must make a specific request in JAL. The logic variable $SETUP is provided to allow you to determine the presence of a /*SETUP control statement.

This parameter does not affect jobs which are exempt from ThruPut Manager processing; the /*SETUP statement receives standard processing for jobs submitted in an Exempted class.

If this keyword is not included, jobs with a /*SETUP statement receive standard JES2 processing.

TYPHOLD

If this keyword is coded, jobs receive ThruPut Manager processing. After Job Analysis the jobs are put in HOLD status. The installation can alter this with JAL. The logic variable $TYPHOLD is provided to allow you to determine whether or not this keyword was present.

If this keyword is not included, jobs submitted with TYPRUN=HOLD receive standard JES2 processing. In this case, the jobs are held prior to ThruPut Manager analysis.

TYPSCAN

If this keyword is coded, ThruPut Manager does JCL analysis for jobs submitted with TYPRUN=SCAN. After the JCL analysis phase, the jobs are not allowed to continue to execution. When errors are found, appropriate messages appear with the JCL listing.

If this keyword is not included, jobs submitted with TYPRUN=SCAN receive standard JES2 processing. In this case, the jobs are not processed by ThruPut Manager, therefore they do not benefit from the JCL analysis provided by the Job Analyzer.

To implement JCL verification, code the appropriate keyword in the TMPARM statement.

Dependencies

There are no other dependencies with this facility.

TMSS Initialization Statements

There are no TMSS initialization statements for this function.

Facilities Summary

TMPARM Statement Refer to *Installation Guide*		
Keywords	**Description**	**Chapter**
SETUP	Requests ThruPut Manager processing for jobs with a /*SETUP statement.	3
TYPHOLD	Requests JCL verification at submission time for jobs submitted with TYPRUN=HOLD.	3
TYPSCAN	Requests ThruPut Manager JCL verification for jobs submitted with TYPRUN=SCAN.	3

Chapter 20. CPU Capping and Normalizing

This chapter describes the facilities provided by ThruPut Manager to limit the CPU amount that any particular job can request from the system. Services are also provided to "normalize" the CPU value across CPUs to reflect different processor speeds.

Description

MVS does not provide any means to limit the total CPU time that a job may request. The only service available is the ability to provide *a default value per step* when the user has not coded CPU step time in the JCL. This presents the installation with some problems:

1. It only takes place when the user has not coded any CPU values. Hence, unless the installation adds code in a system exit to remove CPU values coded by the user, the CPU requirements for any given job are unpredictable. This is true regardless of the job class.
2. More important, the default CPU values are per step. For example, if the default value is 10 seconds a three-step job could run for as long a 30 CPU seconds. A six-step job could consume up to 60 CPU seconds. For the purpose of initiator deployment to meet batch services, this is undesirable, since the execution time of any particular job cannot be predicted.

ThruPut Manager provides a solution to these problems with the "CPU Capping and Normalizing" facility.

CPU Capping

The Job Action Language allows you to specify a Job CPU limit for a particular job or type of job. This facility is similar to the value having been coded on the JOB statement.

☞ ***Please note that the actual limit value is "inserted" in the job prior to the SMF exit for job initiation (IEFUJI).***

The value is applied to a job using the JAL action statement SET with the keyword CPUCAP. For example:

```
SET CPUCAP(0:30)
```

The above statement will put a JOB CPU limit of 30 seconds on a particular job.

CPU Normalization

To complement the CPU Capping facility, ThruPut Manager offers a CPU normalization service. This allows an installation to adjust CPU limits to reflect the differences in CPU speed by different processors in a MAS complex.

With the JAL SET statement, you can request that the CPU values for a particular job or type of job be “normalized”. The normalization factors are CPU dependent and are provided to ThruPut Manager in the JES2 TMPARM statement through the keyword FACTORn. Up to 8 sets of factors can be specified, and the appropriate set can be selected through the JAL statement SET NORMALIZE.

Before the SMF exit for job initiation (IEFUJI) and after the CPU limit is applied, the JOB CPU time and the STEP CPU times are then multiplied by the factors provided in the TMPARM statement described in ***Installation Guide***

To request CPU normalization in JAL, you use the action statement SET and the keyword NORMALIZE. For example:

```
SET NORMALIZE(YES)
```

The above statement indicates to ThruPut Manager that the particular job is to have the CPU values normalized prior to execution.

Assume you have specified more than one set of normalization factors on the TMPARM statement:

```
TMPARM ...
FACTOR1=(2.8 PRD1),(1 PRD2),(2.8 PRD3),(3 PRD4),
FACTOR2=(3.3 PRD1),(2 PRD2),(2.8 PRD3),(3.5 PRD4),
FACTOR3=(4 PRD1),(5.5 PRD2),(5.5 PRD3),(4 PRD4),
...
```

In JAL, you can then select the appropriate set by using the FACTOR keyword on the SET NORMALIZE statement:

```
SET NORMALIZE(YES) FACTOR(3)
```

For a further description of the SET statement, refer to the ***JAL Reference Guide***.

Implementation Summary

To implement CPU Capping, you do not have to make any special provisions. Simply code the appropriate JAL SET statement.

To implement CPU normalization, you must provide the factors to be used in the normalization process:

- The factors are provided with the JES2 TMPARM card described in ***Installation Guide***.
- The JES2 TMPARM statement allows you to specify a factor to be associated with a particular processor by specifying the SMFID of that particular system. You can therefore have a single TMPARM control statement for all the systems associated with a given MAS complex.
- Up to 8 different sets of factors can be supplied through TMPARM and selected through JAL, allowing you to accommodate jobs from widely varying environments. Each set of factors can specify up to 32 systems.
- If normalization is requested but no factors are supplied, no adjustments are made to the CPU values.

TMSS Initialization Statements

There are no TMSS initialization statements associated with this function. Instead, the FACTORS are provided with the JES2 TMPARM statement. The reason for this is simple: it allows you to execute jobs and apply the FACTORS even if the TMSS address space is not active.

TMPARM — The FACTORn Keyword

The FACTORn keyword is part of the ThruPut Manager JES2 initialization statement TMPARM. It has the form:

```
FACTORn={ff.ff | (ff.ff smfid)[,(ff.ff smfid),... ]}
```

where:

n

Is a digit from the range 1-8, uniquely identifying a set of factors.

For the syntax and keyword placement, refer to ***Installation Guide.***

Facilities Summary

TMPARM Statement Refer to *Installation Guide*		
Keywords	**Description**	**Chapter**
FACTORn	Allows you to specify a CPU normalization factor.	3

Chapter 21. Volume Status (VOL) Function

This chapter describes the VOL function and its implementation.

Description

The Volume Status (VOL) Function allows you to “inform” ThruPut Manager of the status of DASD volumes, regardless of their *real physical status*. This information is kept in the ThruPut Manager DASD Volume List. ThruPut Manager adjusts its resource analysis accordingly. Additionally, the contents of the DASD Volume List can influence whether a job is placed in MHS_VOL hold at job initiation, as explained below.

Making better use of this facility requires an understanding of the “volume determination process” done by ThruPut Manager. In its handling of resident DASD volumes, ThruPut Manager distinguishes between situations where the volume serial number has been “hard-coded” in JCL or comes from the catalog. ThruPut Manager’s default handling of the situation is as follows:

1. If the resident volume was “hard-coded” but is not resident, then ThruPut Manager assumes that it is an error (finger-problem in JCL coding). The job is flushed with an appropriate message.
2. If the volume comes from the catalog, ThruPut Manager assumes that the volume is temporarily unavailable. If the volume is still not available at job initiation time, the job is requeued and MHS delays the job by removing its system affinity and placing it in the MHS_VOL category, with a notation indicating which volume(s) caused the delay.

The above processing logic covers a situation where there is no entry for the DASD volume in the ThruPut Manager’s “Volume Status Table”. You can instruct ThruPut Manager to alter its behavior with the Volume Status Function. Facilities are provided for volume status to be communicated to ThruPut Manager through:

- TMSS Initialization Statements.
- Operator Commands.
- An installation Exit.

Implementation Considerations

Once you have implemented a Volume Status Table, ThruPut Manager's behavior changes to reflect your table:

1. If the volume was "hard-coded" but is not resident, then ThruPut Manager looks for the volume serial number in the Volume Status Table.

   a. If the volume is not found in the table, the job is flushed with an appropriate error message.

   b. If the volume is found, ThruPut Manager takes action depending on how the volume is defined in the table, as described below.

2. If the volume comes from the catalog, ThruPut Manager looks for the volume serial number in the Volume Status Table. If the volume is found in the table, the action taken depends on how the volume is defined in the table, as described below.

The volume list allows you to mark volumes as having one of five possible states:

Ignore—This indicates that when this volume entry is coded as a single VOL=SER entry in JCL, it is to be ignored. This replicates the behavior of the Dummy Storage Group in SMS.

Migrate—This indicates that any dataset found with a volume serial number equal to the one specified here is a "migrated" dataset. This allows you to define the ***"trigger volser"*** that your storage manager uses to detect datasets under its control.

Retired—This indicates that this DASD volume was part of the system before but it is no longer available. Any job which refers to a "retired" volume is failed with an informative message. This is useful in situations where you have removed from the system one or more online volumes. There could still be entries in the catalog referring to the retired volume(s).

Resident—This indicates that the DASD volume is to be treated as resident (and already mounted) ***regardless of its real status in the system***. As an example, you might use this to avoid holding jobs that refer to unavailable volumes in conditional steps that will not be executed.

Unavailable—This indicates that a particular DASD volume is not available at this time. As with volumes indicated as resident, ***this is independent of the actual status of the volume***. Upon encountering a reference to a volume in "unavailable" status, ThruPut Manager requeues the job, delays the job by removing its system affinity, and places it in the MHS_VOL category with a notation indicating which volume(s) caused the delay.

Implementation Summary

To set up your ThruPut Manager DASD Volume List:

- Code IGNORE for any volume serial numbers that have been defined as part of the SMS Dummy Storage Group.
- Code RETIRED for volumes that will not be available again.
- If you have a storage management system such as CA-Disk (formerly SAMS:Disk and DMS/OS) or DFHSM, code MIGRATE for the volume serial number that identifies migrated datasets. In the case of DFHSM, it is MIGRAT. In the case of CA-Disk, it is ARCHIV.
- Code RESIDENT in situations where you want a volume treated as RESIDENT regardless of its true status.
- Code UNAVAILABLE for situations where a volume is temporarily out of service. This status is normally managed with operator commands, since it is temporary.

Dependencies

There are no other dependencies to activate this function.

TMSS Initialization Statements

There is one initialization statement associated with the VOL function:

```
VOL SET
```

Facilities Summary

VOL Initialization Statements		
Statement	**Description**	**Page**
VOL SET	Sets the initial status of a volume in the ThruPut Manager DASD Volume List.	86

VOL Operator Commands	
Command	**Description**
VOL ADD	Adds a particular volume to the ThruPut Manager DASD Volume List.
VOL DELETE	Deletes a particular volume from the ThruPut Manager DASD Volume List.
VOL DISPLAY	Displays the volume status in the ThruPut Manager DASD Volume List.
VOL RESET	Resets the status of a volume in the ThruPut Manager DASD Volume List.

VOL EXITS		
Exit #	**Exit Name**	**Description**
Exit 5	Volume Status	This exit receives control every time the Job Analyzer encounters a *unique volume serial number*. Control is given to this exit after the Volume List has been scanned.

Chapter 22. ThruPut Manager (TM) Function

This chapter describes the TM function. Its purpose is to group a number of initialization statements, commands and exits that define environmental parameters for ThruPut Manager.

Description

The ThruPut Manager (TM) function is a catch-all function under which a number of environmental definitions are grouped. For each function there is a corresponding TMSS initialization statement under the function id of TM. A complete description of the statements is found in “Chapter 2. TMSS Initialization Statements.” The different functions are listed below.

ACS Statement

Activates the ACS support and allows you to define the unit names associated with the facility.

ATL Statement

Activates the 5400 ATL support and allows you to define the unit names associated with the facility.

BATCH Statement

Allows you to batch TM commands in a file to be executed at initialization time.

CATALOG SET Statement

Specifies the timeout interval and requeue priority to be used when MVS catalog management is not responding.

EXIT Statement

This statement specifies which installation exits are to receive control from ThruPut Manager, and their initial status.

SMS Statement

This statement informs ThruPut Manager how the DFSMS storage groups are defined in your installation, and which groups are associated with various facilities such as IBM 3495 ATLDS support and VTS.

TRACE Statement

This statement activates and deactivates the ThruPut Manager tracing facility.

UNIT Statement

This statement allows your installation to inform ThruPut Manager of unit name changes that your system may alter after the job has been processed by ThruPut Manager.

USER Statement

This statement defines user areas that can be associated with each job. Space is then provided for you to place data. These data areas can be accessed by installation exits.

Implementation Summary

Each individual statement should be reviewed for its applicability to your installation.

Dependencies

There are no dependencies for the TM function.

TMSS Initialization Statements

All TM functions are associated with TMSS initialization statements, as indicated above.

Facilities Summary

TMSS Initialization Statements		
Statement	**Description**	**Page**
TM ACS	Indicates to ThruPut Manager that the ACS interface is to be used.	66

TMSS Initialization Statements		
Statement	Description	Page
TM ATL	Indicates to ThruPut Manager that the 5400 ATL interface is to be used.	66
TM BATCH	Indicates to ThruPut Manager that a batch of TM commands is to be read from a file and queued for execution.	68
TM BTLS	Indicates the esoteric unit name(s) and library names used for VTS by the IBM 3495 robotic tape library running in Basic Tape Library System (BTLS) mode.	69
TM CATALOG SET	Specifies the timeout interval and requeue priority to be used when MVS Catalog Management is not responding.	70
TM VTFM	Indicates the esoteric and generic unitnames, and the subsystem name needed for VTFM support.	75
TM EXIT	Specifies which installation exits are to receive control and their initial status.	72
TM SMF	Specifies which SMF user record is used to write SMF data, as well as which data are to be collected.	74
TM TRACE	Activates and deactivates the tracing facility.	76
TM UNIT HANDLE	Informs ThruPut Manager of unit name mappings.	77
TM UNIT SET	Informs ThruPut Manager of default unit name.	78
TM USER	Allows you to alter the default values of the user data areas provided.	79
TM VTAPE	Indicates the DFSMS DATACLAS used for virtual tape volumes by CA-Vtape support.	80

TM Operator Commands	
Command	**Description**
TM BATCH	Allows the operator to request the execution of a group of commands stored in a file.
TM CATALOG	Displays or sets the options used by ThruPut Manager when searching MVS catalogs during job analysis.
TM CLASS	Allows the operator to display and alter the classes that are selected or exempted from ThruPut Manager processing.
TM DISPLAY	Displays the current levels of software for ThruPut Manager, JES2, and MVS, or the initialization parameters for ThruPut Manager.
TM EXIT	Specifies which installation exits are to receive control and their initial status. It can also be used to display the status of exits.
TM OPTIONS	Allows you to enable, disable, and display the status of components.
TM TRACE	Activates and deactivates the tracing facility.
TM UNIT	Manages unit name mappings used to inform ThruPut Manager about unit name changes that occur after Job Analysis.

TM Exits		
Exit #	**Exit Name**	**Description**
Exit 10	TMSS Initialization	This exit receives control when the TMSS address space is started.
Exit 11	TMSS Termination	This exit receives control when the operator requests that the TMSS address space terminate processing.
Exit 12	TMSS Command Inspection	This exit receives control when ThruPut Manager Operator Services (TMOS) is about to process an operator command.

The Job Setup Services Component

Chapter 23. JSS Hold Extensions

This chapter introduces you to the job hold extensions available with the TM/JSS Option of ThruPut Manager.

Introduction

JSS needs to be able to determine whether or not a job is being held because its setup requirements have not yet been met. Since the standard JES2 hold function is inadequate for this task, ThruPut Manager *extends* the JES2 system hold.

Since ThruPut Manager *extends* the JES2 hold facility, any standard JES2 hold is completely unaffected. Jobs that have been held by some action not connected with JSS will ***not*** be released when JSS holds are removed.

Automated Hold/Release

Through the architecture of the Control File Manager and the Control File, ThruPut Manager adds several new types of hold to the single JES2 system hold. In the Base Product, an additional hold is provided by Job Setup Services. It is the HSM Time Delay Hold.

HSM Hold

One of the facilities offered by JSS is an early dataset recall for DFHSM migrated datasets. Whenever you request early dataset recall, it is possible to place a time delay on the job to give the system the opportunity to complete the recall. The job is placed in HSM hold until the delay time expires.

The recall mechanism allows you to differentiate between datasets migrated to DASD and datasets migrated to tape.

☞ ***Note that the delay is a "timer" mechanism independent of the actual dataset recall process.***

Control File Considerations

JSS requires the Control File. All the information needed to manage the new types of job hold is maintained in this file. For a description of the Control File, refer to “Chapter 3. File Definition Services (FDS) Function.” For a description of the Control File Manager implementation requirements, refer to ***Installation Guide*** “Chapter 7. Activating the Control File Manager.”

Operational Considerations

JSS provides operator commands to manage the new types of holds. Except for jobs in OPERATOR hold, intervention to release jobs from JSS holds should be rare. Jobs in VOLUME and HSM hold are normally released *automatically*.

End User Considerations—Display Facilities

The User Display Services facility has been extended to allow users to distinguish JSS holds from ordinary JES2 holds. For more information refer to “Chapter 5. User Display Facility (UDF) .”

Chapter 24. JAL Extensions for JSS

This chapter provides a brief description of the JAL extensions that are part of the TM/Job Setup Services Component.

Description

The TM/Job Setup Services component of ThruPut Manager extends the capabilities of JAL to address the management requirements of jobs that require setup. These extensions cover:

- Additional Job Descriptors covering the area of robotic cartridge systems.
- JAL Action statements to give instructions to JSS so it can manage this type of workload as automatically as possible.

This chapter presents a brief description of the JSS JAL extensions. They are fully documented in the manual ***JAL Reference Guide***.

Job Descriptors for Robotic Systems

TM/JSS provide interfaces to the following automated library systems:

- StorageTek ACS System.
- Memorex Telex 5400 ATL System.
- IBM 3495 ATLDS System.

The resource analysis phase of ThruPut Manager makes use of these interfaces to create Job Descriptors such as $UNIT_AUTO_3490E. This Descriptor provides a unit count of 3490E drives that are part of an automated system. The equivalent Descriptor for non-automated 3490E drives is $UNIT_MANUAL_3490E.

JAL Action Statements

TM/JSS adds the following Action statement:

```
JSS RECALL
```

A brief description of this Action statements is included below.

The JSS RECALL statement means that the job currently being analyzed becomes managed by JSS. In this case, the job is placed in HSM hold, and ThruPut Manager initiates a recall for any dataset that has been migrated to archival storage by DFHSM. For a detailed description of this service, see “Chapter 30. Dataset Recall Services (DRS).”

Chapter 25. TM/JSS Support in the TMISPF Application

This chapter provides a short introduction to the TM/JSS extensions to the ISPF-based application provided to support ThruPut Manager.

Introduction

TM/JSS is supported by the TMISPF application. To invoke ThruPut Manager ISPF support, issue the command:

```
TMISPF
```

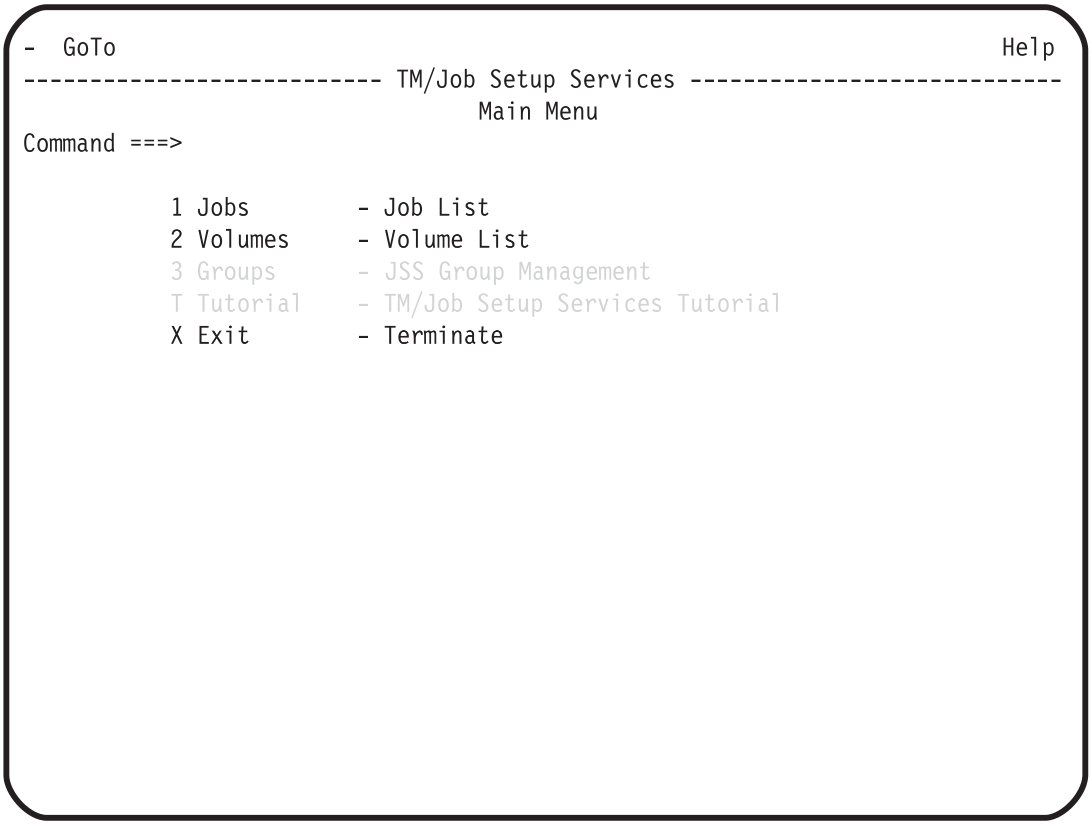
From the TMISPF Main Lobby, select **Applications**, then the **TM/JSS Dialog**. The following capabilities are relevant to TM/JSS:

- The Job List screen shows details about all jobs managed by TM/JSS, including information about each job's volume requirements.
- The Volume List screen provides details about every volume in the Volume Information File.

Note that all the command driven requests provided with TM/JSS can also be requested through the ISPF application.

The TM/JSS Dialog Main Menu

The TM/JSS dialog, selected from the Operating Services menu, displays the TM/JSS Menu, as illustrated:



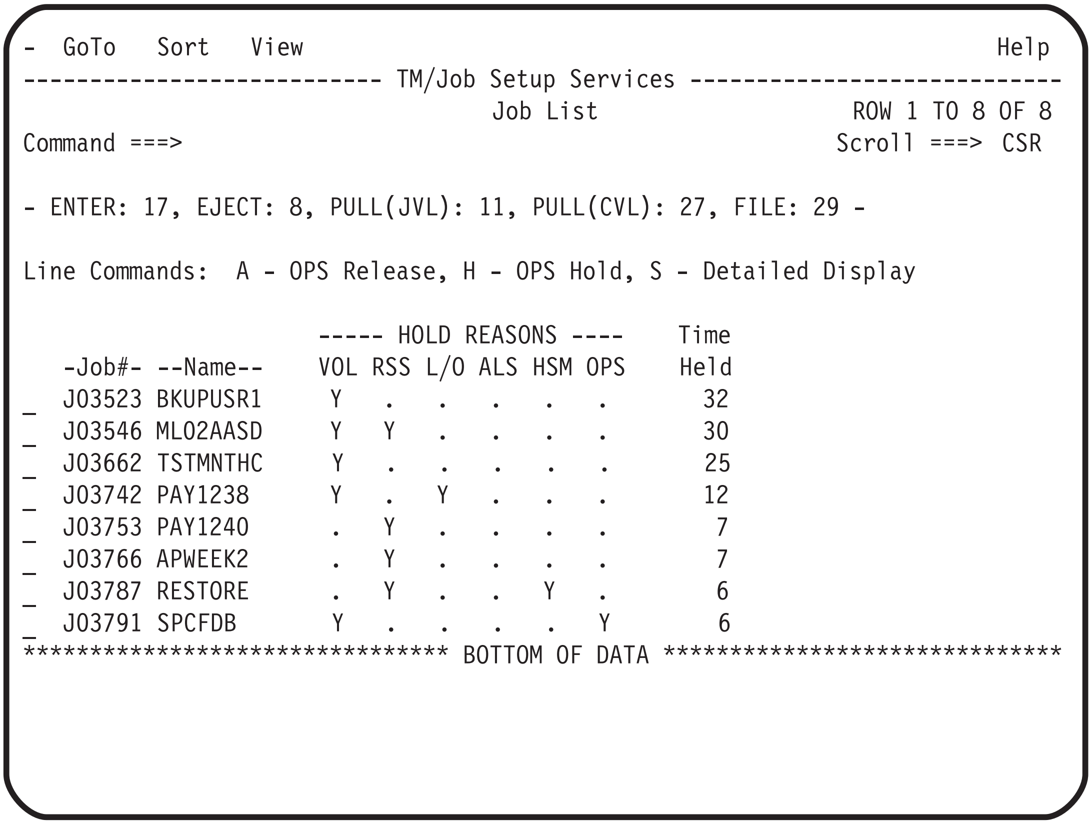
```
-  GoTo                                                                   Help
--------------------------- TM/Job Setup Services ----------------------------
                                  Main Menu
Command ===>

           1 Jobs        - Job List
           2 Volumes     - Volume List
           3 Groups      - JSS Group Management
           T Tutorial    - TM/Job Setup Services Tutorial
           X Exit        - Terminate
```

The JSS Dialog Main Menu

From the TM/JSS Menu, you can choose to display a summary of all jobs or all volumes. You can make your choice by entering the number of the selection you want at the prompt, or by using the drop-down menu from the action bar at the top of the screen.

The Job List Screen

The Job List screen displays a summary of all jobs managed by TM/JSS. Here is an example:



```
-  GoTo   Sort   View                                                      Help
--------------------------- TM/Job Setup Services ----------------------------
                                   Job List                  ROW 1 TO 8 OF 8
Command ===>                                                 Scroll ===>  CSR

- ENTER: 17, EJECT: 8, PULL(JVL): 11, PULL(CVL): 27, FILE: 29 -

Line Commands:  A - OPS Release, H - OPS Hold, S - Detailed Display

                      ----- HOLD REASONS ----    Time
   -Job#- --Name--    VOL RSS L/O ALS HSM OPS    Held
_  J03523 BKUPUSR1     Y   .   .   .   .   .       32
_  J03546 ML02AASD     Y   Y   .   .   .   .       30
_  J03662 TSTMNTHC     Y   .   .   .   .   .       25
_  J03742 PAY1238      Y   .   Y   .   .   .       12
_  J03753 PAY1240      .   Y   .   .   .   .        7
_  J03766 APWEEK2      .   Y   .   .   .   .        7
_  J03787 RESTORE      .   Y   .   .   Y   .        6
_  J03791 SPCFDB       Y   .   .   .   .   Y        6
******************************** BOTTOM OF DATA *****************************
```

The JSS Dialog Job List Screen

This screen provides a line for each job, showing its job number and name, the reasons why the job is held, and the length of time that it has been in hold.

The **Sort** command on the action bar allows you to sort the display using any column as the key. You can also use the **View** command on the action bar to switch between a display of all jobs and a display of held jobs.

Chapter 26. StorageTek Automated Cartridge System (ACS) Function

This chapter provides a description of the interface to the ACS and the logic that ACS Support uses to classify cartridges.

Description

In order to provide accurate unit counts during Job Analysis, ThruPut Manager must know which unit names are used for the StorageTek Automated Cartridge System (ACS), and how they are used. The following description explains this interface.

Implementation Summary

ThruPut Manager provides an interface to the StorageTek ACS. The call to the interface is issued automatically.

Dependencies

You must install the Host Software Component of the ACS at the appropriate level. It must include the Programmatic Interface.

ThruPut Manager assumes that the program name is SLSXCAL, as documented in the ***StorageTek HSC System Programmer's Guide***. It must be available to be called from any address space in your system.

How ACS Support Works

This description does not apply to virtual tape volumes. Please refer to "Chapter 29. Virtual Tape Systems (VTS) Support."

For installations with an ACS, ThruPut Manager provides a means to determine whether or not a given cartridge is ACS managed.

The ACS support function works as follows:

- **At TMSS initialization time,** the ACS Programmatic Interface is called. If the LOAD macro fails, TMSS terminates with appropriate messages. Otherwise, TMSS automatically detects AUTOMATED and MANUAL units.
- **During resource determination**, ThruPut Manager's ACS support uses the following logic to determine when to interrogate the ACS API:
    - For a non-specific request, no ACS call is made:
        - If the unit name matches any of the unit names detected as MANUAL, it is treated as a MANUAL (non-ACS) request.
        - If the unit name does not match any of the unit names detected as MANUAL, it is treated as an AUTOMATED (ACS) request.
    - For specific requests, ThruPut Manager interrogates the ACS interface to determine whether or not the cartridge is resident:
        - If the cartridge is resident, it is treated as an AUTOMATED (ACS) request, ***unless a unit name was coded in the JCL:***
            - If the unit name is detected as MANUAL, the volume is treated as a MANUAL (non-ACS) volume.
            - If the unit name is not detected as MANUAL, the request is treated as an AUTOMATED (ACS) volume.
        - If the cartridge is not resident ***and a unit name was coded in the JCL***, then:
            - If the unit name is detected as AUTOMATED, it is treated as a cartridge to be returned to the Automated Library System.
            - If the unit name is not detected as AUTOMATED, the request is treated as a MANUAL (non-ACS) volume.
        - If the cartridge is not resident ***and a unit name was not coded in the JCL***, it is treated as a MANUAL (non-ACS) volume.
- If failure occurs during Job Analysis, a non-deletable message is issued and the cartridges are treated as manual cartridges. The Job Analyzer continues to interrogate the interface for each cartridge, but no further messages are produced if the interface fails. Once the interface is re-established, the non-deletable message is removed.

Facilities Summary

ACS EXITS		
Exit #	**Exit Name**	**Description**
Exit 4	Unit Name/Device Type	A flag is provided to indicate that an ACS esoteric unit name has been encountered.
Exit 5	Volume Status	A flag is provided to show that ACS has responded to the ThruPut Manager, indicating that the volume is ACS managed.
Exit 6	Job Action Modification	Three ACS related Job descriptors are provided: The maximum number of ACS managed drives, the number of ACS volumes, and the minimum number of ACS scratches.

Chapter 27. Memorex Telex 5400 Automated Tape Library (ATL) Function

This chapter provides a description of the interface to the 5400 ATL and the logic that ATL Support uses to classify cartridges.

Description

In order to provide accurate unit counts during Job Analysis, ThruPut Manager must know which unit names are used for the Memorex Telex 5400 Automatic Tape Library System (ATL), and how they are used. The following description explains this interface.

Implementation Summary

ThruPut Manager provides an interface to the Memorex Telex 5400 Automatic Tape Library System. The call to the interface is issued automatically, and is controlled with the TM ATL initialization statement. In order to make use of this function you have to:

- Install the Library Management Software (LMS) component of the 5400 ATL at the appropriate level.
- Include the TM ATL statements in your ThruPut Manager initialization statements. There must be at least one TM ATL statement for the 5400 ATL interface to be activated. The types of ATL units you can define are:

```
TM ATL AUTOMATED(unit-name list)
TM ATL MANUAL(unit-name list)
TM ATL VTL(unit-name list)
```

Where:

`AUTOMATED`

The unit name is defined as: **“Must always be served by the Automated Library System regardless of where the volume is located.”**

`MANUAL`

The unit name is defined as: **“Must always be served manually regardless of where the volume is located.”**

☞ ***Please note that the ATL unit name matching is only applicable to unit names coded in JCL. The standard z/OS catalog entry does not distinguish ATL cartridges from non-ATL cartridges.***

Also note that the TM definitions are for analysis purposes only. ThruPut Manager does not alter the drive selection algorithms.

Dependencies

The 5400 Library Management Software supporting the API must be installed.

ThruPut Manager assumes that the program name is LMSACCES, as documented in the Memorex Telex Supplemental documentation for the 5400 LMS System. It must be available to be called from any address space in your system.

TMSS Initialization Statements

There is one type of initialization statement associated with this facility: TM ATL. For a description, see “Chapter 2. TMSS Initialization Statements.”

How the ThruPut Manager 5400 ATL Support Works

This description does not apply for virtual tape volumes. Please refer to “Chapter 29. Virtual Tape Systems (VTS) Support.”

For installations with a 5400 ATL, ThruPut Manager provides a means to determine whether or not a given cartridge is LMS/ATL managed.

The ATL support function works as follows:

- **At TMSS initialization time,** the ATL Programmatic Interface is called if the TM ATL statement is included. If the LOAD macro fails, TMSS terminates with appropriate messages.
- **During resource determination,** ThruPut Manager’s ATL support uses the following logic to determine when to interrogate the LMS API and what to do with the information:

- For a non-specific request (scratch), no ATL call is made:
  - If the unit name matches any of the unit names defined with a ‘TM ATL MANUAL’ statement, it is treated as a manual request.
  - If no unit name match occurs, or if it matches a ‘TM ATL AUTOMATED’ statement, it is treated as an automated scratch request.
- For specific requests, ThruPut Manager interrogates the ATL interface to determine whether or not the cartridge is resident:
  - If the cartridge is resident, it is treated as an ATL request, unless a UNIT name was coded in the JCL that matches a ‘TM ATL MANUAL’ statement. If that is the case, it is treated as a manual request.
  - If the cartridge is not resident it is treated as a manual request unless a UNIT name was coded in the JCL that matches a ‘TM ATL AUTOMATED’ statement. If that is the case it is treated as a cartridge to be returned to the ATL system.

- **If a failure occurs during Job Analysis,** a non-deletable message is issued and the cartridges are treated as manual cartridges. The Job Analyzer continues to interrogate the interface for each cartridge, but no further messages are produced if the interface fails. Once the interface is re-established, the non-deletable message is removed.

Facilities Summary

TMSS Initialization Statements		
Statement	**Description**	**Page**
TM ATL	Defines the esoteric unit names for ATL managed drives.	66

ATL EXITS		
Exit #	**Exit Name**	**Description**
Exit 4	Unit Name/Device Type	A flag is provided to indicate that an ATL esoteric unit name has been encountered.

ATL EXITS		
Exit #	**Exit Name**	**Description**
Exit 5	Volume Status	A flag is provided to show that LSM has responded to the ThruPut Manager, indicating that the volume is LSM managed.
Exit 6	Job Action Modification	Three ATL related Job descriptors are provided: The maximum number of ATL managed drives, the number of ATL volumes, and the minimum number of ATL scratches.

Chapter 28. DFSMS Tape Storage Groups (SMS) Function

This chapter provides a description of the interface to DFSMS tape storage groups.

Description

ThruPut Manager provides an interface to DFSMS tape storage groups, such as those affected by IBM Tape Mount Manager (TMM). ThruPut Manager automatically detects the DFSMS storage groups used by your installation. During Job Analysis, if a new DFSMS-managed dataset does not belong to a detected storage group, the dataset is considered a DASD dataset.

Implementation Summary

You might require the Library Management Software (LMS) component of the 3495 Library Dataserver software. See below.

Dependencies

The SMS Function supports the IBM 3495 Automated Tape Library Dataserver by automatically detecting the ATLDS storage groups. ThruPut Manager interrogates the status of cartridges by invoking LCS External Services. To support ATLDS storage groups, you must therefore install the LCS External Interface for the 3495 Library Dataserver.

How the ThruPut Manager 3495 Library Dataserver Support Works

This description does not apply to virtual tape volumes. Please refer to “Chapter 29. Virtual Tape Systems (VTS) Support.”

For installations with a 3495 Library Dataserver, ThruPut Manager provides a means to determine whether or not a given cartridge is managed by the Library Dataserver.

The ATLDS support works as follows:

- **At TMSS initialization time,** the LCS External Interface is invoked if a 3495 Library dataserver is detected. If the interface is not present or active TMSS terminates with appropriate messages.
- **During resource determination,** ThruPut Manager's ATLDS support uses the following logic to determine when to invoke LCS External Services and what to do with the information:
  - For a non-specific request (scratch), no External Services call is made. If the SMS storage group associated with the scratch request matches any of the storage group names detected as AUTOMATED, it is treated as an automated scratch request.
  - For specific requests, ThruPut Manager invokes LCS External Services to determine whether or not the cartridge is resident:
    - If the cartridge is ***library resident***, it is treated as an automated request.
    - If the cartridge is ***shelf resident*** (belongs to the automated library but is outside) then the cartridge is treated as a tape to be returned to the 3495 dataserver.
    - If the cartridge is not known to the library system, then it is treated as a non-automated cartridge.
- If a failure occurs during Job Analysis when attempting to access LCS External Services, a non-deletable message is issued and the cartridges are treated as manual cartridges. The Job Analyzer continues to interrogate the interface for each cartridge, but no further messages are produced if the interface fails. Once the interface is re-established, the non-deletable message is removed.

Facilities Summary

SMS EXITS		
Exit #	**Exit Name**	**Description**
Exit 5	Volume Status	A flag is provided to show that External Services has responded to the ThruPut Manager, indicating that the volume is managed by the automated system.
Exit 6	Job Action Modification	Job descriptors are provided for the maximum number of VTS drives, the number of VTS volumes, and the minimum number of VTS scratches.

Chapter 29. Virtual Tape System (VTS) Support

This chapter describes the support for Virtual Tape Systems (VTS) provided in ThruPut Manager.

Description

In order to provide accurate unit and volume counts during Job Analysis, ThruPut Manager must know which unit names are used for Virtual Tape Systems (VTS). The following description explains this interface.

Implementation Summary

VTS support varies depending on the way your installation manages virtual tape volumes.

- For IBM 3495 Automated Tape Library Tape Dataserver (ATLDS) virtual volumes:
  - New virtual volumes are recognized by the storage group name, as detected automatically by ThruPut Manager.
  - Old virtual volumes are identified by calling the Library Management Software (LMS) interface to determine if the volume is associated with the detected library.
- For BTLS mode for the IBM 3495 ATLDS:
  - New virtual volumes are recognized by the unit name specified by the VTS_UNIT keyword on the TM BTLS initialization statement.
  - Old virtual volumes are recognized by the library name as specified on the VTS_LIBRARY on the TM BTLS initialization statement.
- For StorageTek Automated Cartridge System (ACS) virtual volumes:
  - New virtual volumes are recognized by the unit name, as detected automatically by ThruPut Manager.
  - Old virtual volumes are identified by calling the Programmatic Interface of the ACS Host Software Component.

- For Sutmyn (Memorex) 5400 Automated Tape Library (ATL) virtual volumes:
  - New virtual volumes are recognized by the unit name, as specified by the VTL keyword of the TM ATL initialization statement.
  - Old virtual volumes are identified by calling the ATL Library Management Software interface.
- For VTFM virtual volumes:
  - New virtual volumes are recognized by the unit name, as specified by the ESOTERIC keyword on the TM VTFM initialization statement.
  - Old virtual volumes are recognized by the unit name, as specified by the GENERIC keyword on the TM VTFM initialization statement.
- For CA-Vtape (formerly SAMS:VTAPE) virtual volumes:
  - New virtual volumes are recognized by the DFSMS dataclass as specified by the DATACLAS keyword on the TM VTAPE initialization statement.
  - Old virtual volumes are recognized by calling the CA-Vtape API.

Dependencies

For StorageTek ACS, the Programmatic Interface must be installed. ThruPut Manager assumes that the program name is SLSXCAL, as documented in the ***StorageTek HSC System Programmer's Guide***. It must be available to be called from any address space in your system.

There are no other specific dependencies for ThruPut Manager VTS support.

TMSS Initialization Statements

VTS support requires a TMSS initialization statement, but the specific statement required depends on the method your installation uses to manage virtual volumes:

- IBM 3495 ATLDS and StorageTek ACS virtual volumes managed by SMS are handled automatically by ThruPut Manager.
- StorageTek ACS virtual volumes are also handled automatically by ThruPut Manager.
- IBM 3495 ATLDS virtual volumes managed by BTLS require the TM BTLS initialization statement.

- Sutmyn (Memorex) 5400 ATL virtual volumes require the TM ATL initialization statement.
- VTFM virtual volumes require the TM VTFM initialization statement.
- CA-Vtape virtual volumes require the TM VTAPE initialization statement.

For descriptions of these statements, refer to "Chapter 2. TMSS Initialization Statements ."

How ThruPut Manager VTS Support Works

ThruPut Manager can provide volume and unit counts for VTS once you provide initialization statements that reflect the way your installation manages virtual volumes. These counts are provided through Range Descriptors in JAL:

```
$SCRATCH_VTS
$UNIT_VTS
$VOL_VTS
```

Virtual volumes and units are not included in any other counts, e.g. the Range Descriptor $SCRATCH does not include virtual scratch volumes.

Facilities Summary

TMSS Initialization Statements		
Statement	**Description**	**Page**
TM ATL	Defines the unit names for Sutmyn (Memorex) 5400 ATL support.	66
TM BTLS	Defines unit and library names used for VTS by the IBM 3495 ATLDS in BTLS mode.	69
TM VTFM	Defines the unit names used for VTFM VTS.	75
TM VTAPE	Defines the DFSMS DATACLAS used for CA-Vtape support.	80

Chapter 30. Dataset Recall Services (DRS)

This chapter describes the Dataset Recall facilities of ThruPut Manager.

Description

Installations that have DFHSM, or a product that supports the DFHSM Recall interface (e.g. ABR, CA-DISK), can make use of the Dataset Recall Services function of ThruPut Manager.

Implementation Summary

ThruPut Manager identifies migrated datasets by the volume serial number specified by the MIGRATE keyword of the VOL SET initialization statement. Refer to “Chapter 2. TMSS Initialization Statements.”

ThruPut Manager attempts to determine the type of migrated dataset by checking the device type in the catalog. The results of this check are returned by the Range Descriptors $HRECALL_DASD and $HRECALL_TAPE. If the device type is not used or not supported, the number of migrated datasets is still returned by the Range Descriptor $HRECALL.

The Dataset Recall function is controlled at the job level with JAL.

Dataset Recall Without Holding the Job

If you do not want to have the job held for dataset recalls, do not use the MAX_WAIT or DELAY_FOR keywords. Instead, use:

```
JSS RECALL ALL
```

ThruPut Manager initiates a recall for all datasets that it determines have been migrated and does not apply the JSS hold category HSM to the job.

Dataset Recall Using Hold

If you want ThruPut Manager to apply the JSS hold category HSM to jobs requiring dataset recall and the DFHSM ARCHRCAL interface is supported, use the MAX_WAIT keyword of the JSS RECALL statement:

```
JSS RECALL ALL MAX_WAIT(nnn)
```

where ***nnn*** represents the maximum number of minutes that the job should be held awaiting the dataset(s) to be recalled. The job is placed in the JSS hold category HSM, which is removed when the required datasets have been recalled, or when the specified number of minutes have elapsed, whichever comes first. See below for further considerations.

If the DFHSM ARCHRCAL interface is not supported, you can use the DELAY_FOR keyword:

```
JSS RECALL ALL DELAY_FOR(nnn)
```

where ***nnn*** represents the number of minutes that must elapse before the JSS hold category HSM is removed from the job. ThruPut Managers issues the DFHSM SVC to recall the dataset and applies the hold. The hold is not removed until the full time has expired, even if all the recalls complete first. Once the time expires, the hold is removed regardless of whether the recalls have been completed.

In JAL, use the $HRECALL_DASD and $HRECALL_TAPE Range Descriptors to determine exactly which types of migrated datasets a job requires.

Bypassing Dataset Recall

Certain job steps allocate migrated datasets but never open them. Usually these are steps that execute IEFBR14 or a similar program, and the datasets are allocated simply to allow a normal termination disposition of DELETE. Clearly, it is a waste of resources to do the actual recall for these datasets.

You can bypass these recalls by specifying the TMSS initialization statement JSS SET:

```
JSS SET PROGRAM_NAMES(IEFBR14) BYPASS_RECALLS
```

If a step executes a program named in the JSS SET statement, no recall processing is performed for any migrated dataset that is allocated with a normal termination disposition of DELETE (DISP=(OLD,DELETE), DISP=(SHR,DELETE), or DISP=(MOD,DELETE) where MOD implies OLD). Everything else is the same except that the datasets are not recalled, meaning that the datasets do in fact get deleted.

Refer to “Chapter 2. TMSS Initialization Statements.” for the syntax description of JSS SET.

For example, a job that deals with GDG datasets:

```
//SAMPJOB     JOB
//          ...
//LASTSTEP    EXEC PGM= IEFBR14, COND=(4,LT)
```

```
//DELETE1     DD   DSN=BACKUP(-1),DISP(OLD,DELETE)
//DELETE2     DD   DSN=BACKUP(-2),DISP(OLD,DELETE)
//DELETE3     DD   DSN=BACKUP(-3),DISP(OLD,DELETE)
//DELETE4     DD   DSN=BACKUP(-4),DISP(OLD,DELETE)
//          ...
```

Let’s assume there are 5 generations of BACKUP.
BACKUP(0) is on DASD. The next two generations BACKUP(-1) and BACKUP(-2) are migrated to ML1, which resides on DASD. BACKUP(-3) and BACKUP(-4) are migrated to ML2 which resides on tape. It will take several minutes to retrieve GDG datasets from (possibly multiple) tapes. The step does nothing, but deletes these GDG datasets. Bypassing recalls for these GDG datasets using the TMSS initialization statement mentioned above can save time and resources.

The JSS BYPASS_RECALLS will take effect for ***all*** datasets that have been HSM migrated with the DELETE normal termination disposition in steps executing a listed program, unless overridden by the DAL statement JSS ALLOW HSM_RECALLS.

Dependencies

For dataset recall, your system must support the DFHSM SVC. To use the MAX_WAIT keyword, your system must support the DFHSM ARCHRCAL interface. To bypass dataset recall, your system must support the DFHSM ARCHDEL interface.

DRS Initialization Statements

To use dataset recall, you must indicate the volume serial number that identifies migrated files by coding a VOL SET initialization statement of the following form:

```
VOL SET V(volser) MIGRATE
```

where ***volser*** is the “trigger” volume serial number.

To bypass dataset recalls, you must specify the names of eligible programs by coding a JSS SET initialization statement of the following form:

```
JSS SET PROGRAM_NAMES(program[,program,...]) BYPASS_RECALLS
```

where ***program*** is the program name.

Refer to “Chapter 2. TMSS Initialization Statements.” for more details

Considerations

The actual recall is asynchronous. The JSS hold category HSM is not applied unless you code a MAX_WAIT or DELAY_FOR keyword on the JSS RECALL statement.

Dataset recalls are issued for all datasets that ThruPut Manager determines have been migrated, based on the volume serial number.

JAL provides Job Descriptors indicating the number of datasets that need recalling. You can take appropriate actions based on that information:

- For datasets migrated to DASD, a hold might not be necessary since the recall time is usually shorter than the job's queue time.
- For datasets migrated to tape, or in cases where you are not certain where the dataset has been migrated, it might be useful to specify a reasonable estimate of the time it will take to recall the dataset. Note that the dataset recall is asynchronous. The JSS RECALL method used determines when the JSS hold category HSM is removed from the job.
  - For MAX_WAIT, the JSS hold category HSM is removed when the dataset recall completes or the time expires, whichever comes first.
  - For DELAY_FOR, the JSS hold category HSM is not removed until the time expires, even if the dataset recall completes earlier.

  In both cases, the JSS hold category HSM can be removed early though an operator command.

If a job is to be re-analyzed, as when using Deferred Processing Services, you should make sure that you do not request a recall until the re-analysis occurs.

Facilities Summary

TMSS Initialization Statements		
Statement	**Description**	**Page**
JSS SET	Specifies the names of programs that are eligible to bypass dataset recall processing.	60
VOL SET	The keyword MIGRATE indicates the volume serial that identifies migrated files.	86

DRS Operator Commands	
Command	**Description**
JSS ADD	Adds a program name to the list of programs eligible to bypass dataset recalls.
JSS DELETE	Removes a program name from the list of programs eligible to bypass dataset recalls.
JSS DISPLAY	Displays the list of programs eligible to bypass dataset recalls.
VOL ADD	The keyword MIGRATE indicates the volume serial that identifies migrated files.
VOL DELETE	Can be used to remove the MIGRATE volume that signals which files have been migrated.
VOL DISPLAY	Can be used to see if a given volume is defined as MIGRATE in the ThruPut Manager DASD Volume List.
VOL RESET	Can be used to alter the status of a volume to MIGRATE, if it is already present in the ThruPut Manager DASD Volume List.

VOL EXITS		
Exit #	**Exit Name**	**Description**
Exit 5	Volume Status	This exit receives control every time the Job Analyzer encounters a *unique volume serial number*. Control is given to this exit after the Volume List has been scanned. If the volume matches the “migrated” volume, a flag is on at the time your exit receives control.

The SYSOUT Services Component

Chapter 31. SYSOUT Services (SOS) Function

This chapter introduces SYSOUT Services and uses examples to illustrate the concepts and power of SOS.

Description

SYSOUT Services (SOS) provide facilities for you to use DAL (Detail Action Language) to examine the characteristics of DD statements describing SYSOUT datasets, and to change them if desired. There is also some limited facilities available in JAL to deal with global values.

The Problem

To manage printing services for batch jobs can be a challenging task. As the flexibility to make requests using traditional JCL increases, so do management problems. Setting up standards, communicating them to users, and ensuring that they are followed can be difficult.

Changes to JCL streams are costly and error prone. Further, when you want to make changes there is little room to experiment. Your users are probably (and understandably) reluctant to modify JCL. If you force them to spend time making changes only to find out that the chosen approach is less than ideal, well... good luck if you ask for changes again.

Changes, however, are often necessary. The reasons are many, and vary from installation to installation. The most common ones are:

- New printing devices are installed. In some cases, this requires new work selection rules.
- User groups are restructured in ways that require consolidation or separation of printing tasks.
- User groups are physically relocated, requiring changes in destination parameters.
- JES2 nodes are consolidated within a datacenter. This is a common occurrence as the topology of JES2 and the SYSPLEX are made congruent.
- Datacenters are consolidated. This is normally a challenging project. It is exceptionally rare for the “merging” datacenters to have the same standards. In most cases, you must develop a new set of standards to accommodate heterogeneous JCL streams.

SOS deals with all of these issues. Since the requirements are fluid and different in every case, SOS lets you alter ***anything*** that a user can request with a DD statement containing a SYSOUT keyword, an OUTPUT JCL statement, or the “old fashioned” JES2 /*OUTPUT statement. SOS makes all the complex interrelations of these statements transparent to the person implementing the changes.

The “alterations” could relate to parameters as common as CLASS or as obscure as PRTOPTNS. Of course, MSGCLASS is also one of the values you can modify. You can also negate the effects of an OUTPUT JCL statements or add a new one. Refer to ***DAL Reference Guide*** and ***JAL Reference Guide*** for a complete list.

The Solution: SOS

SOS is a function of the Base Product ThruPut Manager. Its power comes from:

- The ThruPut Manager Job Analyzer facilities.
- The full capabilities of the Job Profile and Action Language facilities of ThruPut Manager.
- Extensions to JAL/DAL that provide a simple but effective architecture to deal with the complex relationships of DD statements with the SYSOUT keyword (referred to as SYSOUT DD), OUTPUT JCL statements, /*OUTPUT JES2 statements, /*ROUTE PRINT statements, and system defaults.

The DAL facilities and architecture give you complete control over job streams with a minimum of technical staff effort. As with all ThruPut Manager facilities, the best way to demonstrate the power and simplicity of SOS is with examples.

Some Hypothetical Job Streams

Here are two hypothetical jobs, taken from examples of SYSOUT processing in an IBM ***JCL User’s Guide*** manual.

```
//BILL002A JOB    ,'DEPT. 25',USER=ACCT003
//OUT1     OUTPUT COPIES=1,DEST=COMPLEX1
//OUT2     OUTPUT COPIES=2,FORMS=A,DEFAULT=YES,DEST=COMPLEX7
//STEP1    EXEC   PGM=COLLECT
//OUT3     OUTPUT DEFAULT=YES,COPIES=1,DEST=COMPLEX7
//INPUT    DD     DSN=BILL,DISP=SHR
//MFK1     DD     SYSOUT=A
//MFK2     DD     SYSOUT=B,OUTPUT=*.OUT1
//
```

JCL for Example 1

Example 1

This example shows an explicit reference to an OUTPUT JCL statement. Note that with an explicit reference, all default OUTPUT JCL statements are ignored. In this example:

- The system processes the output from DD statement MFK1 using the options on the OUTPUT statement OUT3 because:
  - MFK1 does not contain an OUTPUT parameter, and
  - OUT3 contains DEFAULT=YES and precedes MFK1 in the same step.
- MFK1 cannot implicitly reference the job-level default statement OUT2 because of the step-level default statement OUT3. If STEP1 had not contained OUT3, MFK1 would have referenced statement OUT2.
- The system processes the output from DD statement MFK2 according to the processing options on the job-level OUTPUT JCL statement OUT1 because DD statement MFK2 explicitly references OUT1 using the OUTPUT parameter.
- Note that the system ignores the processing options on all default OUTPUT JCL statements (OUT2 and OUT3) for MFK2.

```
//BILL003A JOB    MSGCLASS=A,USER=ACCT005
//OUT1     OUTPUT DEFAULT=YES,DEST=COMPLEX7,FORMS=BILLING,
// CHARS=(A0A,A0B),COPIES=2
//OUT2     OUTPUT DEFAULT=YES,DEST=COMPLEX1,...
//STEP1    EXEC   PGM=ORDERS
//R1       DD     SYSOUT=A
//R2       DD     SYSOUT=B
//STEP2    EXEC   PGM=BILLING
//OUT3     OUTPUT DEFAULT=YES,DEST=COMPLEX7,...
//B1       DD     SYSOUT=A
//B2       DD     SYSOUT=B,OUTPUT=(*.OUT3,*.OUT2)
//STEP3    EXEC   PGM=REPORTS
//OUT4     OUTPUT FORMS=SHORT,DEST=COMPLEX1,...
//RP1      DD     SYSOUT=B
//RP2      DD     SYSOUT=A,OUTPUT=(*.STEP2.OUT3,*.OUT1)
//
```

JCL for Example 2

Example 2

This example shows how the position of the OUTPUT JCL statement affects the processing of the SYSOUT data sets.

In STEP1, the system processes DD statements R1 and R2 using the processing options specified on job-level OUTPUT JCL statements OUT1 and OUT2 because:

- DEFAULT=YES is specified on OUTPUT JCL statements OUT1 and OUT2, and
- There is no OUTPUT JCL statement with DEFAULT=YES within STEP1, and
- The OUTPUT parameter is not specified on DD statements R1 and R2.

In STEP2, the system processes DD statement B1 using the processing options specified on OUTPUT JCL statement OUT3 because:

- DEFAULT=YES is specified on OUTPUT JCL statement OUT3 and OUTPUT JCL statement OUT3 is within the job step STEP2, and
- The OUTPUT parameter is not specified on DD statement B1, and
- OUTPUT JCL statement OUT3 is within STEP2; therefore, the system ignores the DEFAULT=YES specification on job-level OUTPUT JCL statements OUT1 and OUT2 when processing DD statement B1.

In STEP2, the system processes DD statement B2 using the processing options specified on OUTPUT JCL statements OUT3 and OUT2 because:

- Both of the OUTPUT JCL statements are explicitly referenced from the SYSOUT statement. Explicitly-referenced OUTPUT JCL statements can be in any previous procedure or step, before the DD statement in the current step, or at the job-level.
- Note that default OUTPUT JCL statement OUT1 is ignored when processing the data set defined by DD statement B2 because B2 explicitly references OUTPUT JCL statements OUT3 and OUT2.

In STEP3, the system processes DD statement RP1 using the output processing options specified on the job-level OUTPUT JCL statements OUT1 and OUT2 because:

- DEFAULT=YES is specified on OUTPUT JCL statements OUT1 and OUT2, and
- No OUTPUT JCL statement with DEFAULT=YES is coded within STEP3, and

- The OUTPUT parameter is not specified on DD statement RP1.

Note: In STEP3, OUTPUT JCL statement OUT4 is not used at all because it does not have DEFAULT=YES coded, and no DD statement explicitly references OUT4.

In STEP3, DD statement RP2 is processed using OUTPUT statements OUT3 and OUT1. You can explicitly reference an OUTPUT JCL statement in another step if you use a fully qualified reference, such as the reference to OUTPUT statement OUT3 used on DD statement RP2.

You can explicitly reference an OUTPUT JCL statement with DEFAULT=YES coded, such as the reference to OUT1 from DD statement RP2. The system ignores the DEFAULT parameter and uses the remaining processing options according to the normal rules that apply when coding explicit references.

Altering the Job Streams

Altering the above JCL streams (or any other) to make adjustments for SYSOUT dataset printing services can be done in one of three ways:

- Of course, you can always modify the actual JCL.
- You can implement JES2 Exit 40.
- In some cases, you can achieve the desired effects by changing the definitions of workload selection rules for different printers.

Regardless of the method used to alter SYSOUT processing, you *must* understand the relationships. You can make only simple changes without considerations for potential “side effects.”

SOS provides an approach that isolates changes, so the person implementing the alterations does not have to consider any indirect consequences. The concept is simple:

- The SYSOUT DD statement is the anchor. It represents the mechanism for the creation of data. A SYSOUT DD statement represents the basic report. *It produces results whether or not an OUTPUT JCL statement is present.*
- The OUTPUT JCL statements are viewed as distribution requests. *By themselves, they mean nothing.*
- SYSOUT DD statements cannot be added or deleted “externally” to job streams. Creating a “new” SYSOUT requires programming changes.
- OUTPUT JCL statements can be added or deleted “externally”. They do not require programming changes.

This is a simple case. Note that:

- OUT2 is an irrelevant statement.
- This cannot be established by looking at the statement in isolation.
- Any changes to that statement will produce no results.

Now let us look at the second example. Here things get a little more interesting.

```
//BILL003A JOB    MSGCLASS=A,USER=ACCT005
//OUT1     OUTPUT DEFAULT=YES,DEST=COMPLEX7,FORMS=BILLING,
// CHARS=(AOA,AOB),COPIES=2
//OUT2     OUTPUT DEFAULT=YES,DEST=COMPLEX1,...
//STEP1    EXEC   PGM=ORDERS
//R1       DD     SYSOUT=A
//R2       DD     SYSOUT=B
//STEP2    EXEC   PGM=BILLING
//OUT3     OUTPUT DEFAULT=YES,DEST=COMPLEX7,...
//B1       DD     SYSOUT=A
//B2       DD     SYSOUT=B,OUTPUT=(*.OUT3,*.OUT2)
//STEP3    EXEC   PGM=REPORTS
//OUT4     OUTPUT FORMS=SHORT,DEST=COMPLEX1,...
//RP1      DD     SYSOUT=B
//RP2      DD     SYSOUT=A,OUTPUT=(*.STEP2.OUT3,*.OUT1)
//
```

JCL for Example 2 (Repeated)

Example 2

Again using the SYSOUT DD anchor concept, we have the following SYSOUT datasets. From R1:

- R1-01. This is the result of the following pair:

  ```
  R1 DD SYSOUT
  OUT1 OUTPUT JCL
  ```

- R1-02. This is the result of:

  ```
  R1 DD SYSOUT
  OUT2 OUTPUT JCL
  ```

Next, from R2:

- R2-01 as a result of

```
R2 DD SYSOUT
OUT1 OUTPUT JCL
```

- R2-02. as a result of:

```
R2 DD SYSOUT
OUT2 OUTPUT JCL
```

Next, from B1:

- B1 as a result of:

```
B1 DD SYSOUT
OUT3 OUTPUT JCL
```

Next, from B2:

- B2-01 as a result of:

```
B2 DD SYSOUT
OUT3 OUTPUT JCL
```

- B2-02 as a result of:

```
B2 DD SYSOUT
OUT2 OUTPUT JCL
```

Next, from RP1:

- RP1-01 as a result of:

```
RP1 DD SYSOUT
OUT1 OUTPUT JCL
```

- RP1-02 as a result of:

```
RP1 DD SYSOUT
OUT2 OUTPUT JCL
```

Next, from RP2:

- RP2-01 as a result of:

```
RP2 DD SYSOUT
OUT3 OUTPUT JCL
```

- RP2-02 as a result of:

```
RP2 DD SYSOUT
OUT1 OUTPUT JCL
```

A few things to notice:

- There is a total of 4 OUTPUT JCL statements and 6 SYSOUT DD statements.
- 11 SYSOUT datasets will be generated.
- The OUTPUT statement OUT4 is irrelevant.
- Sometimes the same OUTPUT statement is implicitly and explicitly referenced.

The number of permutations and combinations that can be created in any particular job stream is endless, and the scope of any particular OUTPUT JCL statement is extremely difficult to determine for any “large” job.

A More Complex Example

For illustration, let us assume that you want to make the following changes to any job that has a name starting with the characters “BILL” and has a RACF USER id starting with “ACC”:

- Any SYSOUT dataset with a destination of COMPLEX7 is to change to COMPLEX3.
- Any SYSOUT dataset with CLASS B is to have only 1 COPY.
- Any SYSOUT dataset with CLASS B is to have a destination of COMPLEX1.

You probably know how much effort it is to do this in your installation without the help of SOS. Let us show you how easily you would do this *with* SOS.

In a ThruPut Manager environment with the SOS option, you create DAL for SOS to handle the above situation. In the definition section of SOS DAL, you use the definition statements shown here.

```
EVALUATE     JOB_TO_CHANGE      ($JOBNAME(BILL*) & ($RACFU(ACC*))
EVALUATE     IGNORE_JOB         (JOB_TO_CHANGE)
EVALUATE     B_CLASS            ($SOS_CLASS(B))
EVALUATE     WRONG_DEST         ($SOS_DEST(COMPLEX1))
EVALUATE     DEST_COMPLEX7      ($SOS_DEST(COMPLEX7))
$SOS_COPIES  SINGLE_COPY,2,MULTIPLE_COPIES
```

Sample SOS DAL Definition Section

In this example:

- The first statement identifies jobs that you are interested in changing.
- The second statement identifies jobs to eliminate from SOS processing.
- The third statement identifies SYSOUT datasets with a class of B.
- The fourth statement identifies SYSOUT datasets with a destination other than COMPLEX1.
- The fifth statement identifies SYSOUT datasets with a destination of COMPLEX7.
- The sixth statement identifies SYSOUT datasets with a request for multiple copies.

Now, to request the changes, you code the following logic and action statements:

```
IF (IGNORE_JOB)
   RETURN DONE
ENDIF

IF (B_CLASS)
   IF (WRONG_DEST)
      SOS ALTER DEST(COMPLEX1)
   ENDIF
   IF (MULTIPLE_COPIES)
      SOS ALTER COPIES(1)
   ENDIF
RETURN CONTINUE
ENDIF

IF (DEST_COMPLEX7)
   SOS ALTER DEST(COMPLEX3)
ENDIF
RETURN CONTINUE
END
```

Sample SOS DAL Logic Section

This is all there is to it, unless you want to be informative and tell the job submitter about all the wonderful changes that you are making. You can do this with the message facilities of DAL. SOS issues a message informing the user which statements your DAL has altered, unless you turn the feature off:

```
DTM4474I THE FOLLOWING SYSOUT DD STATEMENTS HAVE BEEN ALTERED: 7,8,...
```

Below is a complete SOS DAL, including informative messages.

```
PROC ID(DAL) TYPE(SOS) DESC('SAMPLE SOS DAL')
EVALUATE     JOB_TO_CHANGE ($JOBNAME(BILL*) & $RACFU(ACC*))
EVALUATE     IGNORE_JOB    (JOB_TO_CHANGE)
EVALUATE     B_CLASS       ($SOS_CLASS(B))
EVALUATE     WRONG_DEST    ($SOS_DEST(COMPLEX1))
EVALUATE     DEST_COMPLEX7 ($SOS_DEST(COMPLEX7))
$SOS_COPIES  SINGLE_COPY,2,MULTIPLE_COPIES

MSGDEF COPIES_ALTERED ('SYSOUT STMT# ',$JCL_STMT#,' AND OUTPUT JCL ',   +
                      'STMT# ',$SOS_OUTPUTJCL_STMT#,' COPIES ',         +
                      'CHANGED FROM ',$SOS_COPIES,' TO 1 BECAUSE OF CLASS=B')
MSGDEF DEST_ALTER_TO1 ('SYSOUT STMT# ',$JCL_STMT#,' AND OUTPUT JCL ',   +
                      'STMT# ',$SOS_OUTPUTJCL_STMT#,' DESTINATION ',    +
                      'CHANGED FROM ',$SOS_DEST,' TO COMPLEX1 BECAUSE OF+
                       CLASS=B')
MSGDEF DEST_ALTER_TO3 ('SYSOUT STMT# ',$JCL_STMT#,' AND OUTPUT JCL ',   +
                      'STMT# ',$SOS_OUTPUTJCL_STMT#,' DESTINATION '     +
                      'CHANGED FROM ',$SOS_DEST,' TO COMPLEX3')

IF (IGNORE_JOB)
   RETURN DONE        /*SOS DAL IS NOT CALLED AGAIN FOR THIS JOB */
ENDIF
IF (B_CLASS)
   IF (WRONG_DEST)
      SOS ALTER DEST(COMPLEX1)
      WTU DEST_ALTER_TO1
   ENDIF
```

Sample of Complete SOS DAL (1 of 2)

```
   IF (MULTIPLE_COPIES)
      SOS ALTER COPIES(1)
      WTU COPIES_ALTERED
   ENDIF
   RETURN CONTINUE
ENDIF
IF (DEST_COMPLEX7)
   SOS ALTER DEST(COMPLEX3)
   WTU DEST_ALTER_TO3
ENDIF
RETURN CONTINUE
END
```

Sample of Complete SOS DAL (2 of 2)

Results of SOS Processing

In the following examples, we show certain OUTPUT JCL statements inside boxes. These statements are intended to illustrate the results of DAL processing. They do not actually appear in the JCL listing for the jobs.

Let us look at the two sample jobs SOS presents to the system after it finishes processing. The first job is a simple case:

```
//BILL002A JOB    ,'DEPT. 25',USER=ACCT003
//OUT1     OUTPUT COPIES=1,DEST=COMPLEX1
//OUT2     OUTPUT COPIES=2,FORMS=A,DEFAULT=YES,DEST=COMPLEX7
//STEP1    EXEC   PGM=COLLECT
//$TM0001  OUTPUT COPIES=1,DEST=COMPLEX3
//OUT3     OUTPUT DEFAULT=YES,COPIES=1,DEST=COMPLEX7
//INPUT    DD     DSN=BILL
//MFK1     DD     SYSOUT=A,OUTPUT=*.$TM0001
//MFK2     DD     SYSOUT=B,OUTPUT=*.OUT1
//
```

JCL for Example 1, Showing Results of SOS Processing

In this case, only one OUTPUT JCL statement has to be created.

The second job is more interesting, and is illustrated below. Please note that this is shown here to give you a conceptual understanding about the workings of SOS. The actual internal implementation details shown here need not concern the implementer of SOS rules.

As before, the created OUTPUT JCL statements are shown inside boxes.

```
//BILL003A JOB    MSGCLASS=A,USER=ACCT005
//OUT1     OUTPUT DEFAULT=YES,DEST=COMPLEX7,FORMS=BILLING,
//                CHARS=(AOA,AOB),COPIES=2
//OUT2     OUTPUT DEFAULT=YES,DEST=COMPLEX1,...
//STEP1    EXEC   PGM=ORDERS
//$TM0001  OUTPUT DEST=COMPLEX3,FORMS=BILLING,
//                CHARS=(AOA,AOB),COPIES=2
//$TM0002  OUTPUT DEST=COMPLEX1,...
//$TM0003  OUTPUT DEST=COMPLEX1,FORMS=BILLING,
//                CHARS=(AOA,AOB),COPIES=1
//$TM0004  OUTPUT DEST=COMPLEX1,...
//R1       DD     SYSOUT=A,OUTPUT=(*.$TM0001,*.$TM0002)
//R2       DD     SYSOUT=B,OUTPUT=(*.$TM0003,*.$TM0004)
//STEP2    EXEC   PGM=BILLING
//OUT3     OUTPUT DEFAULT=YES,DEST=COMPLEX7,...
//$TM0005  OUTPUT DEST=COMPLEX3,...
//$TM0006  OUTPUT DEST=COMPLEX1,...
//$TM0007  OUTPUT DEST=COMPLEX1,...
//B1       DD     SYSOUT=A,OUTPUT=*.$TM0005
//B2       DD     SYSOUT=B,OUTPUT=(*.$TM0006,*.$TM0007)
//STEP3    EXEC   PGM=REPORTS
//OUT4     OUTPUT FORMS=SHORT,DEST=COMPLEX1,...
//$TM0008  OUTPUT DEST=COMPLEX1,FORMS=BILLING,
//                CHARS=(AOA,AOB),COPIES=1
//$TM0009  OUTPUT DEST=COMPLEX1,...
//$TM0010  OUTPUT DEST=COMPLEX3,...
//$TM0011  OUTPUT DEST=COMPLEX3,FORMS=BILLING,
//                CHARS=(AOA,AOB),COPIES=2
//RP1      DD     SYSOUT=B,OUTPUT=(*.$TM0008,*.$TM0009)
//RP2      DD     SYSOUT=A,OUTPUT=(*.$TM0010,*.$TM0011)
//
```

JCL for Examples 2, Showing Results of SOS Processing

Implementation Summary

SOS is part of the Base Product ThruPut Manager, therefore there are no special implementation procedures required. Here is a summary of some of the areas to consider when dealing with SOS.

System-managed Datasets

As indicated above, the architecture of SOS is based on the idea that the SYSOUT DD statement is the anchor that drives the DAL process. There is an exception to this rule.

The OUTPUT JCL statement allows you to control the printing characteristics of the "system-managed datasets": the JCL listing, the LOG, and the system messages for the job. There is no corresponding SYSOUT DD statement for each of these listings. SOS handles this situation by "driving" DAL once for each JES dataset to be generated.

A Unique variable, $SOS_SYSOUT_DS, indicates whether or not you are processing a system-managed SYSOUT dataset or a regular SYSOUT dataset. If it is a system-managed SYSOUT dataset, then you can also determine which type. You can alter the characteristics of any one of them.

*/*OUTPUT JECL Statement*

In the examples, the /*OUTPUT statement is not shown. SOS handles job streams that include that type of statement alone or in conjunction with OUTPUT JCL statements. The most obvious difference in the way the system processes OUTPUT JCL and /*OUTPUT JECL statements is in the "override" parameter considerations.

In general, with OUTPUT JCL statements the SYSOUT DD has the last word. The opposite is true with the /*OUTPUT JECL statement. SOS solves this as follows:

- The SYSOUT DD statement becomes a skeleton statement with a minimum of parameters.
- The parameters are all transferred to the inserted OUTPUT JCL statement. This takes place regardless of the "pair" being processed. That is, the same construct takes place for a (SYSOUT DD)/(OUTPUT JCL) as for a (SYSOUT DD)/(/*OUTPUT) pair. All the alterations result from an inserted OUTPUT JCL statement.

Removing SYSOUT Datasets

The deletion (or negation) of a SYSOUT dataset can be accomplished by setting the OUTDISP to (PURGE,PURGE). SOS provides a simpler technique, however: the SOS ALTER REMOVE statement.

Adding SYSOUT Datasets

The creation of a new SYSOUT dataset is the result of inserting an additional OUTPUT JCL statement. You can also insert additional OUTPUT JCL statements by re-driving SOS DAL. The RETURN statement in SOS DAL supports the REDRIVE keyword, which causes ThruPut Manager to re-invoke SOS DAL for the current SYSOUT dataset. The COUNT keyword repeats the re-driving until a specified number of OUTPUT JCL statements have been inserted.

SOS Logging

By default, SOS records (in the system messages for the job) the SYSOUT DD statement numbers for which alterations were requested. In this way there is an indication that changes have taken place. You can turn off this logging with the statement SOS SET DO_NOT_LOG_ALTERS.

If you want to provide detailed information about the changes, use the message construct facilities of SOS.

Dependencies

SOS is integrated into ThruPut Manager and has no other dependencies.

Facilities Summary

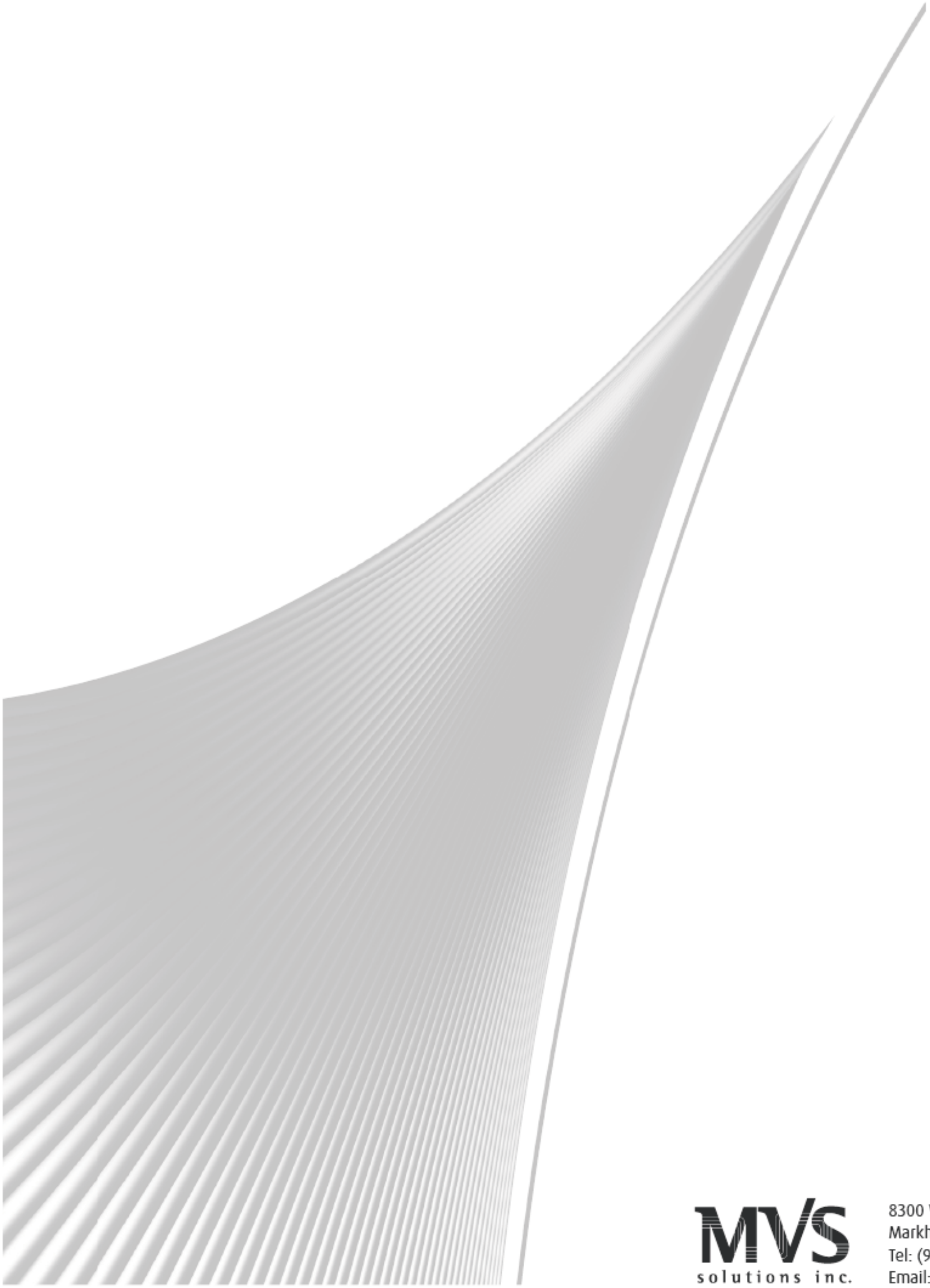
SOS Initialization Statements		
Statement	Description	Page
DAL LOAD	Specifies the dataset name and member name (if applicable) where the SOS DAL internal text is located.	43
DAL TRACE	Allows you to request DAL trace at TMSS start up time. You can trace specific jobs or all jobs.	44

SOS Operator Commands	
Command	**Description**
DAL DISPLAY	Displays the active DAL and its identifier.
DAL REFRESH	Allows you to activate a new DAL dynamically.
DAL TRACE	Allows you to activate tracing for all jobs, or selected jobs.

SOS JECL Statements		
Statement	**Description**	**Page**
/*DAL TRACE	Activates DAL tracing for this job.	

Index

Index



MVS
solutions inc.

8300 Woodbine Avenue, 4th Floor
Markham ON Canada L3R 9Y7
Tel: (905) 940-9404 Fax: (905) 940-5308
Email: marketing@mvssol.com www.mvssol.com