CA SOLVE:FTS

Administration Guide

Release 12.1

CA Technologies Product References

This document references the following CA Technologies products:

- CA SOLVE:FTS

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

This section contains the following topics:

Intended Audience

This guide is intended for technical personnel responsible for the planning, setup, and maintenance of your product's functions and services.

Typographic Conventions

This table explains the conventions used when referring to various types of commands and when indicating field attributes.

Convention	Description
Commands	Commands such as SYSPARM and SHUTDOWN are shown in uppercase.
User Entries	Information to enter onto panels is displayed in **bold** text.
Cross-References	Cross-reference links to other sections of the book are displayed as underlined blue text.
Shortcuts	Shortcuts to menus or options are displayed in **bold**, for example, **/PARMS**.

Chapter 2: Planning the File Transfer Environment

This section contains the following topics:

Getting Started

CA SOLVE:FTS moves bulk data between an installation's computer sites by using the resources of the network that links those sites together.

CA SOLVE:FTS should be regarded as a potential high-volume user of the available network capacity. Consequently, you should consider the manner in which it is used in your organization and the installation options that are available. This ensures that the introduction of CA SOLVE:FTS traffic into the network does not have adverse effects on existing batch or interactive operations.

Operational Planning

Privileges available to a user are included in their user ID definition (see page 37). CA SOLVE:FTS provides the required definition panels for use in the Userid Access Maintenance System (UAMS) and support for the relevant structured fields if a full security exit is installed to replace the UAMS component.

The allocation of privileges to users is likely to be determined by the manner in which CA SOLVE:FTS is used in your installation, and the implications of allocation of the various privileges are reviewed here.

Allocation of System Privileges

System privileges let authorized users define, request, and control the transmissions of production files, and to have overall control of the system.

Consider the following points when deciding to whom you want to authorize private privilege:

- System definition privilege lets users set up transmission definitions that, when executed, cause CA SOLVE:FTS to assume responsibility for access to the data sets involved, rather than cause it to consider the access as personal access by the user issuing the transmission request.

  Therefore, a user who is assigned system definition privilege should not also be assigned system request privilege. This prevents one individual user from being able to define the requirement for access to sensitive data sets and to cause CA SOLVE:FTS to carry out that access.

  Note: Additional security is available by restricting such users to specific terminals.

- System request privilege lets users request transmission of system definitions as dictated by the system access mask associated with their user ID. System request privilege does not imply private request privilege, but the user may have private request privilege too.

  System request privilege is usually restricted to Operations.

  Note: Additional security is available by restricting such users to specific terminals.

- System control privilege lets users maintain overall control of the system. All monitoring and control functions are available, including control over the initiator definitions for all destinations.

  System control privilege is usually restricted to Operations and systems programmers responsible for support.

Allocation of Private Privilege

Private privilege lets authorized users move files from one location to another.

Consider the following when deciding to whom you want to authorize private privilege:

Private Definition Privilege

You can allocate the Private Definition privilege to many users to let them set up their own transmission definitions on the VSAM database. You should enforce naming conventions and limit the number of definitions that any one user may set up. You can impose these limitations by doing *one* of the following:

- Select suitable private access masks.
- Allocate the Private Definition privilege to only a few users who can then set up private definitions on request from the remaining users.

Private Request Privilege

You can assign the Private Request privilege to many users to let them schedule for transmission, at any time, any private definition that they may access (as dictated by their private access mask).

If private users are allowed to use any available transmission class, this may allow private transmissions to preempt, or run at a higher priority than, system transmissions. Restriction of private transmission requests to installation specified classes provides control over the number of initiators that are available for servicing private transmissions. This control in turn can be used to limit the number of private transmissions that may be in progress to a particular destination at one time, and allows all private activity to be suspended at peak system transmission times. The access authorization exit can be written to enforce classes for private users or you can assign the Private Request privilege to Operations, who can then attend to the requirements of the remaining users.

Private Control Privilege

You can let all users monitor, interrupt, and restart their own private requests. In this scenario, users must also look after their own transmissions, thereby removing the need for operations supervision of private requests. Alternatively, you can let Operations supervise private transmissions.

Initiators and Transmission Classes

The processing of a transmission request is performed by a function called a transmission initiator. A set of 16 initiators is maintained for each remote system to which transmissions can take place.

Each initiator services one or more transmission classes (A to Z) and each transmission request has a transmission class assigned when it is defined. Consequently, a given transmission request can be processed only by a transmission initiator that services its transmission class. The initiator/transmission class concept is similar to the standard operating system initiator and job class mechanism, and is designed to be used in the same manner.

Decide the Number of Initiators in Each Set

You need to consider how many initiators in each initiator set you want to define for each target system and whether certain types of transmission are assigned to specific transmission classes.

The system default, in the absence of specifically defined initiator sets, is to provide a single active initiator that services all transmission classes for each remote system. If transmissions are infrequent and of short duration, this default may be sufficient; however, if there are many transmissions to be scheduled, of varying priorities and duration, it is necessary to adopt appropriate standards for the scheduling of different requests under different classes.

Capacity Planning Considerations

The implementation of CA SOLVE:FTS in a network usually implies the movement of bulk data across a telecommunications network rather than between channel-linked CPUs only. The bandwidth available between the various sites determine the speed that files are transmitted. Capacity planning involves considerations of available bandwidth, operational requirements for multiple concurrent transmissions, and the effect of high-volume batch data on existing network traffic.

Pace Traffic

When a transmission is in progress, CA SOLVE:FTS does not attempt to pace the rate of data transfer. For example, if a transmission is occurring across a T1 link, in the absence of other network traffic, CA SOLVE:FTS drives the link at a very high percentage of its capacity.

Because such a network link is likely to carry interactive traffic in addition to CA SOLVE:FTS traffic, it is important to prevent CA SOLVE:FTS traffic from interfering with interactive response times. Data flow control in the network is a function of the network software being used and various techniques are available to reduce the priority and impact of high volume batch traffic passing through the network.

When using VTAM as the underlying transport mechanism, you can allocate a low transmission priority to CA SOLVE:FTS INMC sessions by using an appropriate Class of Service table entry. Alternatively, you can use VPACING on the APPL statements to pace the traffic, although this can lead to severe degradation in performance.

Use Staging Data Sets

You can transmit any sequential data set using dynamic allocation; however, it is sometimes advisable to use staging data sets at the transmitting or receiving end of a transmission, or possibly both.

At the transmitting host, a *staging* data set describes a data set into which data is copied, or staged, in preparation for transmission. The priming of the staging data set can be performed by using a system utility or an application program. If a staging data set is used at the receiving end, it implies that the data must be copied out, again using a utility or application program, prior to that staging data set being used again for receipt of a subsequent transmission.

Various mechanisms and utilities are provided to aid in the use of staging mode transmissions.

Project Groups

Often the need arises for a discrete area or project group to transmit their own data sets. Rather than let each individual access all facilities, you can allocate to a project group one or more data sets into which data to be transmitted can be copied. The actual transmission can be performed by a person outside of the group (for example, a central operator) on request.

After the data set is transmitted to the remote location, the data can be copied into the relevant target data set.

Safeguard Shared Staging Data Sets

There are various ways in which a staging data set can be shared by a number of users. Each user can write their own data into the data sets using DISP=MOD so that it is appended to the end of the file. Alternatively, a scheme can be set up where you can enter data into the staging data set only if it is empty.

CA SOLVE:FTS can empty a data set on successful transmission, and a utility is provided to ensure that a data set is empty before copying into it, thus facilitating the implementation of such techniques.

Transmission Definition Requirements

When you use staging mode, you must set up safeguards to prevent inadvertent overwriting of an input staging data set before successful transmission of a previous one, and to prevent receiving a transmission into a staging data set that has not been emptied from a previous transmission. Specific options on the definition of transmissions are provided to assist in safeguarding staging data sets.

When defining the From data set details, you can empty that data set when it has successfully transmitted. This, together with the restriction that data must not be copied into it unless it is empty, prevents the inadvertent overwriting of a data set before it is transmitted. A utility is provided to check that a data set is empty.

When defining the To data set details, you can set it so that the transmission cannot proceed unless the output data set is empty. This, together with a procedure that empties an output staging data set after the data in it has successfully copied out, prevents the inadvertent overwriting of an output data set. A utility is also provided that empties a data set.

Utilities

Many batch utility programs are supplied in the source and load distribution libraries (in *dsnpref.*NMC1.CC2DLOAD). You can change these utilities as required.

This section details the basic functions of each utility and notes the applicable operating system.

You need to study the documentation included in each utility (in *dsnpref.*NMC1.CC2DSAMP). The documentation contains information about runtime requirements and restrictions, such as warnings about using the utilities with partitioned data sets.

UTIL0001: Batch Job Interface

This utility submits any operating system commands supplied through JCL PARM or SYSIN data file. Its primary use is to provide a means for a batch job to request a file transmission. You can do this by requesting the utility to submit a Modify command of the following format:

```
F NM,TRANSMIT transdefa
```

To CA SOLVE:FTS, this Modify command looks like it was entered by an operator from the system console, and therefore looks like it is sourced from the SYSOPER (or alternative installation-defined) console user ID. TRANSMIT is an Operator Console Services (OCS) command requesting transmission of the file defined, in this case, the *transdefa* transmission definition.

You can include a step that executes this utility in a production run and use it to automate the transmission of files created in earlier job steps.

UTIL0002: Staging File Copy

This utility can be executed in place of IEBGENER to copy a data set into a staging data set prior to transmission. It ensures that the DCB attributes of the staging data set are made the same as the data set being copied into it, and thus eliminates abnormal termination of IEBGENER due to conflicting DCB attributes.

UTIL0003: File Empty Check

This utility verifies that a data set is empty by checking that a staging data set from which previous data was transmitted is empty before copying data into it. If any previous transmission from the staging data set has completed successfully, it is emptied (if the EMPTY INPUT option was selected when the definition was created), and copying proceeds.

The utility usually runs as the step before UTIL0002.

UTIL0004: File Reset

This utility empties any data set by emptying a receive staging data set after the data in it has been copied elsewhere. Once empty, the next transmission into the staging data set can take place.

This utility is designed to work in conjunction with UTIL0003 to protect staging data sets.

UTIL0005: JCL Submission

This utility is called from a Network Control Language (NCL) procedure, using the &CALL statement. It can be passed to JCL statements, and submits them to the operating system internal reader.

As an alternative to using UTIL0005, you can do *one* of the following:

- Use the Data Set Services SUBMIT function, $DSCALL OPT=SUBMIT.

  Note: For more information, see the *Network Control Language Reference Guide.*

- Dynamically allocate a SYSOUT file to the z/OS internal reader and then use &FILE ADD statements to submit the JCL.

Multiple Concurrent Transmissions

When multiple transmissions are in progress between a pair of systems, the files are multiplexed across the INMC link between the two systems to make full use of the bandwidth available. This shares the available bandwidth equally between the transmissions that are taking place, so that the more transmissions running concurrently to a particular destination, the longer each individual transmission takes.

Although initiators can be set to high priority, which expedites their transmissions over standard priority ones, these transmissions are competing for bandwidth with any other transmissions that are running.

Planning should include consideration of the number of transmissions to one destination that can effectively be run concurrently. While it is true that one transmission runs fastest by itself, operationally it is not necessarily most efficient to wait for the end of one transmission before starting the next. For example, if a report file transmission is in progress and will take four hours to complete, there is no reason why other small transmissions should not be run at the same time, rather than waiting until the report transmission completes. You can multiplex concurrent transmissions.

Set Up as a Stand-Alone Product

If you are using CA SOLVE:FTS as a stand-alone product, that is, without any CA NetMaster products, there are restrictions such as defining UAMS information manually because groups are not defined.

Set Up Example

In the following example, an installation has several computer centers linked together by a corporate network. There is a large central site and the remaining sites are smaller and provide more localized data processing facilities. The installation has set up its CA SOLVE:FTS operational environment as follows:

- CA SOLVE:FTS is installed at all sites in the network. Each site has one or more operational personnel assigned system privileges to allow them to monitor and control all CA SOLVE:FTS activity taking place in their site. Only these users can modify the CA SOLVE:FTS environment (for example, modify number of initiators, initiator classes, and so on) at their site.
- These users also have OCS authority and communicate with their CA SOLVE:FTS counterparts in other sites directly from their OCS screens. This provides a private communication system across the network so that they can communicate with each, resolve errors, and so on.
- Reports that are part of the regular data processing operation are prepared at a central site in the network, and then transmitted to the other sites for printing and local distribution. In addition, the other sites transmit data to the central site at periods during the day for inclusion in an overnight update operation.
- Maintenance of both system software and application systems used in the smaller sites is supported only from the central site. Changes to software and applications in the form of updated execution libraries are usually transmitted directly to the target site to eliminate the inconvenience and additional expense of transporting tapes.
- The processing capacity of certain sites is used largely at night, with relatively low usage during the day. The installation therefore provides remote TSO facilities in the remote site to applications development staff located at the central site. From time to time, these staff need to be able to transport working libraries and data sets between the two locations.

Private Users

The following types of private CA SOLVE:FTS users can operate in this network:

- Support and maintenance personnel
- Users of TSO facilities in remote sites

Support and Maintenance Personnel

These users are responsible for maintenance and support of systems in remote sites. They need unrestricted access to facilities. They sometimes may want to transmit data sets at short notice or as an urgent requirement. These users have been given privileges that allow them to define their own transmission requests and to issue those requests at any time.

Users of TSO in Remote Sites

These users are usually members of a development project team. Individually, they use CA SOLVE:FTS on an infrequent basis; therefore, they do not each require CA SOLVE:FTS privileges. However, because their total intermittent requirements amount to regular usage of CA SOLVE:FTS facilities by the project team as a whole, a single member of the team can request a particular private transmission request, which has been defined for the project team.

The project team owns a data set in each site and any information transmitted for a member of the team is copied to the project data set first, and then transmitted. It is the responsibility of the team members to schedule who transmits and when they transmit.

System Users

The following categories of system users are defined:

- CA SOLVE:FTS operators
- Personnel who maintain system definitions

System Operators

The operators are all given system request privileges and they all have a system access mask that allows unrestricted access to any system request. None of the operators have any private privilege because none of them have any requirement to initiate their own personal transmissions, preventing them from requesting anyone else's private transmissions. They do, however, have unrestricted authority over active private requests.

The operators monitor activity and can modify initiators, interrupt transmissions, restart failed requests after the cause of failure has been eliminated, and so on.

Definition Maintenance Personnel

There is a small group of users who are allocated system definition privilege. These people are part of an operational support group and are responsible for setting up the system definitions that are used to effect the transmissions associated with ordinary operations. These users are not usually allowed any other privilege, in particular, system request privilege. This isolates the authority to set up a system transmission request definition from the authority to request CA SOLVE:FTS to execute such a definition, which prevents any one individual from getting CA SOLVE:FTS to access and transmit a production data set.

Transmission Classes

The transmission classes serviced by the transmission initiators to any destination are organized by priority and category of transmission request. Initiators may be allocated high or normal priority. Initiator classes may change during the day as transmission requirements change.

Example: Transmission Classes

The following shows a typical set of initiators to a particular destination.

Initiator	Classes	Priority	Used for
1	J, K, L	High	High priority production transmissions
2	A, B	Normal	Report transmissions
3	A, B	Normal	Report transmissions
4	A, B, C, D, E	Normal	Reports and general production transmissions
5	P	Normal	All private transmissions
6–16	–	–	Unused

Chapter 3: Starting and Stopping a Region

This section contains the following topics:

Start a Region

To start a region, you run it as a job or a started task. A started task has been set up during the installation process.

To start a region, issue the following command:

```
S rname,REUSASID=YES
```

Users log on to a region by using the user IDs and passwords specified in their UAMS (or external security package) records.

Stop a Region

If you have the necessary authority, you can shut down the region.

To stop a region, issue the operating system STOP (P) command.

You can also stop a region by issuing *one* of the following commands: **SHUTDOWN** or **FSTOP**.

SHUTDOWN Command

The SHUTDOWN command stops the region when the last user logs off. When you issue the SHUTDOWN command, a broadcast is issued to all users. No further logons are accepted until the region is restarted, or the SHUTDOWN CANCEL command is issued.

You can issue the SHUTDOWN command from OCS or Command Entry. Alternatively, you can issue it as a z/OS MODIFY command.

Note: For more information about the SHUTDOWN command, see the online help.

FSTOP Command

The FSTOP command immediately disconnects user sessions and shuts down the region.

Restrict the use of the FSTOP command.

You can issue the FSTOP command from OCS or Command Entry. Alternatively, you can issue it as a z/OS MODIFY command.

Important! If you are running another product in the same region, it also stops if the FSTOP command is issued.

Note: For more information about the FSTOP command, see the online help.

How You Preserve Data When Region Stops and Restarts

You can preserve some data when a region stops so that this data is available when the region restarts. You can use global variables to preserve data. You can save global variables that the region reloads when it restarts. Saved global variables are known as persistent global variables.

To preserve data, create global variables with data you want to preserve and save them, for example:

- Use the Persistent Variables Administration option (access shortcut is /PVARS).
- Call the $CAGLBL procedure using the SAVE option.

  Note: For information about the $CAGLBL procedure, see the *Network Control Language Reference Guide*.

Create Persistent Global Variables Using the User Interface

You can create persistent global variables from the Persistent Variables List panel. The panel also lets you maintain those variables, for example, update, purge, or reload them.

To create a persistent global variable using the user interface

1. Enter the **/PVARS** panel shortcut.

   The Persistent Variables List panel appears.

2. Press F4 (Add).

   The Persistent Variable - Add panel appears.

3. Specify the name of the variable (without its global prefix) and its value. Press F3 (File).

The variable is saved so that it can be loaded the next time the region starts up.

Prevent the Reloading of Preserved Data

If problems occur during region startup because of invalid data being loaded, you can disable the reloading of the preserved data.

To prevent the reloading of preserved data, enter the following command when you start the region:

```
S rname,PARM='XOPT=NOPVLOAD'
```

The region starts without reloading the preserved data.

Chapter 4: Configuring a Region

This section contains the following topics:

Region Configuration

After you have completed installation and startup, your region is operational at a basic level; however, you must configure it to suit your requirements.

How You Use JCL Parameters to Configure a Region

JCL parameters enable you to configure a region. You use JCL parameters to set region information. This information includes, for example, the names of your INIT and READY procedures, and the types of security exit to use in your region.

This information is supplied by the PPREF statements in the RUNSYSIN member.

You can also pass this information in the START command using the JCL PARM field. If you specify multiple parameters, separate each with a comma.

Note: For more information, see the *Reference Guide*.

How You Display and Change JCL Parameter Settings

You can display the current settings of all the JCL parameters with the SHOW PARMS command from OCS or Command Entry. To change any of these parameters, specify their new values in the RUNSYSIN member and then restart the region.

Note: For more information about JCL parameters, see the *Reference Guide*.

How You Identify the Region to Users

If you have multiple regions or communicate with other regions, you can set the domain ID and put titles on the panels.

How You Identify Domains and Panels

The NMDID JCL parameter identifies the domain ID for each region. If you have multiple regions, specify a different domain ID for each one.

Note: For more information about the NMDID parameter, see the *Reference Guide*.

You can use the SYSTEMID (System Identifications) parameter group in Customizer to help identify your regions. This parameter group specifies a system identifier that is used when you link to other regions. Specify a different system identifier for each of your regions.

This parameter group also specifies the titles to display on the logon panel and the OCS console panel. These titles help users to identify the region that they have logged on to.

Note: The system ID parameter takes effect when the region is initialized.

Region Customizer

Customizer lets you review and update parameter groups.

You use Customizer to initialize and customize your region. Customizer is an initialization facility that lets you implement a region rapidly and easily. Also, Customizer enables you to customize parameters easily at a later stage.

When you first install a product, you set various parameters to get the product up and running. Customizer helps you set up these parameters. An initial dialog is supplied for the first time user, to walk you through the customization process. You are prompted to supply required and optional parameter values.

To access the parameter groups, enter **/PARMS**.

What Are Parameter Groups?

System parameters are grouped by category (such as Security) in logical parameter groups, to simplify the process of initializing and customizing a region.

Groups of individual parameters translate into one or more of the following:

- SYSPARMS that determine how your region functions
- Global variables that various NCL applications use to control their functions
- Local parameters that define how to implement actions associated with parameter groups

Update System Parameters

Most customization of your region is performed by using Customizer.

You can also use the SYSPARMS command to customize your region. Each operand of the SYSPARMS command lets you specify options to change and customize the way your region works.

Notes:

- SYSPARMS set by Customizer parameter groups can only be updated using Customizer.
- For SYSPARMS without a corresponding parameter group, set the SYSPARMS in the INIT and READY procedures so that they are applied when the region starts, and then update them dynamically using the SYSPARMS command.
- CA SOLVE:FTS provides additional operands on the base SYSPARMS command.
- For more information about SYSPARMS operands, see the *Reference Guide*.

Use the SYSPARMS Command

To change a SYSPARMS operand with the SYSPARMS command, enter the following command at the OCS command line:

```
SYSPARMS operand=value operand=value ...
```

Example: Display Time on OCS Title Line

This example sets the time display at the beginning of the OCS title line using the following command:

```
SYSPARMS OCSTIME=YES
```

Initialization Operands

There are some SYSPARMS command operands that cannot be changed while the region is operational. These operands must be included in your INIT procedure so that they are executed during initialization.

Note: For a complete list of SYSPARMS commands, see the *Reference Guide*.

If you specify new values for these initialization operands, the new values do not take effect until the region is initialized. All other SYSPARMS can be changed during region operation by authorized users.

Product-related SYSPARMS Command Operands

Products supply their own SYSPARMS command operands in addition to those that apply to System Services. These additional operands control the customization of the features provided by those products and are not available if the system is not configured for those products.

Check INMC Link Definitions

CA SOLVE:FTS uses the services of the INMC component to communicate with other CA SOLVE:FTS regions running in your network.

To display the INMC link definitions that define the other regions with which this region can communicate, enter **/INMC** at the prompt. The INMC : Link Status List opens. Check that the link definitions show the status as ACTIVE for all other regions with which CA SOLVE:FTS is to communicate.

Alternatively, you can enter a **SHOW LINK** command from Operator Console Services (OCS) to display the INMC link definitions. Check that the available link definitions define all other regions with which this region is to have CA SOLVE:FTS communications.

Link Regions

As a prerequisite to any activity between two regions, there has to be an active INMC link between those regions and therefore there has to be an INMC link definition available to each region.

To establish a link between two CA SOLVE:FTS regions

1. Issue the following command on the local region (NM01):

   ```
   LINK START=NM02
   ```

   where NM02 is the VTAM application name of a remote region with CA SOLVE:FTS running.

   Example output:

   ```
   N32909 NM02 NOT DEFINED ON VFS, ATTEMPTING SIMPLE-MODE DEFINITION
   N32901 APPLID DEFAULTED TO NM02
   N32905 MSGID DEFAULTED TO NM02
   N32904 LINK DEFINITION FROM NM01 TO NM02 COMPLETE, ACTIVATION PENDING
   ```

2. Issue the following command on the remote region:

   ```
   LINK START=NM01
   Example output:
   N32909 NM01 NOT DEFINED ON VFS, ATTEMPTING SIMPLE-MODE DEFINITION
   N32901 APPLID DEFAULTED TO NM01
   N32905 MSGID DEFAULTED TO NM01
   N32904 LINK DEFINITION FROM NM02 TO NM01 COMPLETE, ACTIVATION PENDING
   N35008 LINK FROM NM02 TO NM01 STARTING
   N35002 LINK FROM NM02 TO NM01 ACTIVE
   ```

3. Issue **SHOW FTS** to display the available links. You can use these links to create file transmission definitions.

   Example output:

   ```
   N42601 Dest  Request  Cl  -Block Count-  --Char Count--   Elap-Time
   N42602 NM02     IN-CONTACT      ENABLED IDLE
   N42603  Outbound:
   N42603  Inbound:
   N42608 *END*
   ```

Note: For more information about the INMC component and INMC link definition requirements, see the *Reference Guide*.

Initiator Requirements

When an INMC link becomes active for the first time, a set of 16 transmission initiators reserved for the transmission of files is built.

Each initiator is assigned one or more transmission classes and has a status of STOPPED or STARTED. Each file transmission operation to the other CA SOLVE:FTS region uses one initiator for the duration of the transmission, so the number of concurrent transmissions that can take place is limited to the number of defined and started initiators.

File transmissions are assigned a transmission class (alphabetic, A to Z) that determines which initiators the transmission may use. The meaning of these classes is similar to that of operating system job classes—that is, the class of a transmission governs the choice of initiators that are able to service that transmission. The installation may therefore separate testing and production, or high and low priority transmissions by class.

Initiator Definitions

You can use the default settings for a group of initiators for a particular definition and CA SOLVE:FTS builds an initiator set configured as follows:

  1 : Services classes A - Z in order. Initial status is `STARTED'.
2-15 : Service class A only. Initial status is 'STOPPED'.

If this default set of initiators is sufficient, you need take no further action. If, however, you want to set up a different set of initiators for this particular destination, you must define the initiator set (see page 33).

When you start the INMC link between the two regions, if an initiator is defined, it is used; otherwise, the default settings are used.

Define Custom Initiators

To define custom initiators

1. Enter **/FTSADMN** at the prompt.

   The FTS : Administration Menu appears.

2. Select option A.

   The FTS : Transmission Initiator Definition panel appears.

3. Complete the following fields:

   Destination name

   Specifies the destination. This is the INMC link name associated with the other end of the transmission request.

   Initially enabled?

   Specifies whether FTS activity is allowed automatically when the INMC link to this system becomes active.

   Maximum concurrent inbound files

   Specifies the maximum number of transmissions this system can receive at any one time from the destination system.

   Service Classes

   Specifies the FTS classes serviced by this initiator.

   Started?

   Specifies the initial status of the initiator.

   Com

   Specifies the compression level.

   Hi-Pr?

   Specifies the priority of this initiator in relation to all other initiators for this destination.

   Pref-Session

   Specifies the preferred INMC session number to use for this initiator.

   Press F4 (Save)

   The definition is saved.

4. (Optional) Issue a **LINK STOP** command to close the link, then a **LINK RESET** command followed by a **LINK START** command. This refreshes the link definition and the initiator definitions, and activates the new initiator definitions.

Example: Initiator Definition

The following example shows an initiator set with five initiators defined, servicing 12 different classes between them.

```
PROD-------------- FTS : Transmission Initiator Definition --------------------
Command ===>                                                  Function=ADD

  Destination name ... NEW YORK          Initially enabled? ... Y
  Maximum concurrent inbound files
Init   Service Classes (A to Z)        Started?   Com   Hi-Pr?   Pref-Session
   1   XY                                Y        3     Y
   2   CFAB                              Y        3     N
   3   AB                                Y        3     N
   4   D                                 Y        3     N
   5   PQRST                             Y        3     N
   6
   7
   8
   9
  10
  11
  12
  13
  14
  15
  16
 F1=Help      F2=Split     F3=File      F4=Save
                           F9=Swap                          F12=Cancel
```

When multiple initiators are defined and a request is available for execution with a transmission class that is serviced by more than one (available) initiator, the request is scheduled under the initiator that has the highest preference for that class.

For example, if none of the five initiators shown are busy and a Class A transmission request is available for execution, CA SOLVE:FTS finds that initiators 2 and 3 both service class A, but that initiator 2 accepts class A requests only if no class C or F requests are available. Initiator 3, however, services class A in preference to any other class. In this case, CA SOLVE:FTS assigns the request to initiator 3 for execution. A second class A request issued while initiator 3 is executing the first request would be allocated to initiator 2.

Check File Transmission Activation

After you have defined all your initiator sets for your first system, you must check the file transmission activation.

To check the file transmission activation

1. Start a second system and define the initiator set for that system.
2. Log on to one system in OCS Mode (ensure that the user ID you use has Monitor Status and CA SOLVE:FTS privileges) and, if necessary, execute a **LINK STOP** command to break the INMC link between the two systems.

   The INMC link becomes inactive.
3. Issue a **LINK START** command to reestablish the link and watch the sequence of monitor messages that appear. For example:

   ```
   LINK START=remote-system
   ```

   The link becomes active and the following monitor message appears:

   N35002 LINK FROM *this-system* TO *remote-system* NOW ACTIVE
4. Enter **SHOW FTS**.

   The status of your CA SOLVE:FTS connection across the link appears.

   a. If the message display shows that CA SOLVE:FTS is IN CONTACT for the link, CA SOLVE:FTS has been activated successfully for this pair of systems and file transmission operations are now possible.

   b. If the display shows PENDING CONTACT, the remote region may not be active or may not allow INMC links to this region.

   c. If the display shows NOT CONFIGURED, the remote system is not configured with the CA SOLVE:FTS function. If the display shows NOT-COMPATIBLE, the remote system is running a different version of CA SOLVE:FTS. If the two versions are incompatible, contact Technical Support.
5. If an INMC link was already active prior to updating the initiator set, you must restart these links.
6. Issue a **LINK STOP** command to close the link, then a **LINK RESET** command followed by a **LINK START** command. This refreshes the link definition and the initiator definitions, and activates the new initiator definitions.

Check Initiator Settings

After you activate CA SOLVE:FTS between the two systems, you must check the initiator settings now in force for the remote system.

You can change the settings; however, changes are temporary and do not affect the saved definition. This panel can be used to make temporary changes to initiator status or classes to meet changing operational needs. The saved settings are reinstated when the region is initialized or the INMC links are re-established.

To check the initiator settings

1. Enter **/FTSIS** at the prompt.

   The FTS : Initiator Supervision Menu appears.

2. Select option 1 and enter the name of the remote system.

   A panel appears that contains the settings of the initiator set currently in use by the system for the specified destination.

   If you defined a set of initiators to be used by CA SOLVE:FTS for this destination system, the displayed panel should be the same as your definition.

Execute Test Transmissions

At this stage, CA SOLVE:FTS is ready for use. You should now set up some test transmissions to familiarize yourself with the method of operation. For information about how to define and request a transmission, see the *User Guide*.

You are advised to run test transmissions (in a test environment) involving different device types (for example, disk to tape), different file record formats, different block sizes, and so on, to become familiar with the various facilities and to introduce errors such as forcing out-of-space conditions to observe the operational aspects of CA SOLVE:FTS under these conditions. If you plan to transmit files between different operating systems, for example, z/OS to VM, you should also check the minor operational differences between the two systems.

One method of checking transmission restarts is to issue a **LINK STOP** command when transmissions are in progress, followed by a **LINK START** to reactivate the link. This will simulate a link failure. Remember, however, that this will also disrupt any ROF sessions that are active on that INMC link.

Set Up Private Users and System Users

In all cases, control over the operation of CA SOLVE:FTS in an organization depends heavily upon the number of users that are allowed access and the privileges that those users are given. Access is allocated by user ID, and the various access privileges are described in the following sections.

Functions fall into the following major categories:

- Defining transmission requests
- Issuing transmission requests
- Controlling activity

Private Definition Privilege

When defining a transmission request, you can classify a user as being allowed to define private definitions or system definitions.

A user authorized to define private definitions is entitled to create, modify or delete transmission request definitions on the region's VSAM database, where the names of those definitions are controlled by an access mask associated with the user's user ID. This access mask can be used to limit the user's range of definition names according to installation standards or requirements.

For example, a user's mask of USER01* lets the user access transmission definitions starting with USER01, with the * signifying that the rest of the 12 character definition name may contain any other valid characters.

Alternatively, a mask of ****Z limits the user to accessing definition names of five characters in length that end in Z and with any other combination of characters in the first four positions.

The use of access masks allows transmission definition naming standards to be enforced according to installation requirements.

The default mask value requires definitions accessible to the user to start with their user ID and be followed by any other combination of characters.

When a transmission definition is filed in the VSAM database in which all CA SOLVE:FTS definitions are maintained, the definition is attributed a status of system or private. The status that is applied depends on the definition privilege assigned to the user filing the definition. This attribute cannot be overridden and is not a definition parameter.

Transmission requests that specify execution of a private definition may be issued only by users privileged to issue private transmission requests. When such a transmission request is issued, the access authorization user exit is driven, requesting authorization for personal access to the data sets involved in the transmission by the user that issued the request.

System Definition Privilege

System definitions are assumed by CA SOLVE:FTS to represent transmissions of a production nature (for example report files, off-site backup data sets, data collection files, and so on), which are used in the normal operational processing of the installation. A system definition is assigned the system attribute because it is a definition filed by a user who has system definition privilege assigned to their user ID. The names of the system definitions accessed by a user are limited by their system definition access mask.

The authorization access user exit is driven by CA SOLVE:FTS when a system definition is selected for transmission, specifying that the region requires authorization to access the data sets involved in the transmission.

A user that has authority to create or modify system definitions is not also allowed to define private definitions. This allows complete isolation of the maintenance aspect of system definitions to a specific user ID. It should be noted that the effect of system definition privilege is that the user is entitled to specify that CA SOLVE:FTS will take responsibility for accessing the data sets involved in a transmission.

Important! The assignment of system request privilege to the same user may cause a security exposure by allowing one individual effective personal access to any data set that CA SOLVE:FTS is authorized to transmit. You should not assign both system definition and system request privilege to any users without good cause.

Private Request Privilege

When issuing a transmission request, you may allow a user to request the transmission of private requests, where the names of the definitions whose execution may be requested are limited by the same access mask as described above.

The request for execution of a private definition is regarded by CA SOLVE:FTS as a request for personal access to the two data sets involved in the definition. The system will drive the authorization access exit specifying the data sets involved and supplying the user ID of the individual that requested execution of the definition. The installation-supplied exit may therefore determine whether the individual should have access to the data sets in question, and if not, may indicate that the transmission should not proceed.

Private request privilege can be held in conjunction with or independently of private or system definition privilege.

System Request Privilege

A user with system request privilege is entitled to request execution of system definitions according to the restrictions imposed by their system access mask. System request privilege is usually allocated only to CA SOLVE:FTS operators involved in scheduling transmissions associated with production processing.

System request privilege may be held in conjunction with private request privilege. There are different access masks to control the system and private definitions that the user is allowed to request.

Private Control Privilege

By using control privilege, you can allow a user limited control over private requests defined by their private access mask. The control allowed includes the monitoring of the requests' progress or status, and the ability to interrupt or modify the status of the private requests. Private control privilege does not allow access to system requests or the various control panels such as initiator supervision.

System Control Privilege

A user that is allocated system control privilege is allowed full control over any private requests regardless of their private access mask, and control over those system requests that match the user's system access mask. System control allows full access to all supervisory panels. This privilege level is usually reserved for system operators.

System control privilege is required for operational aspects of CA SOLVE:FTS such as changing the number of initiators active for a particular destination or the request classes being serviced by those initiators.

Although this privilege level allows operational control of the system, the user's ability to modify the status of system requests is still limited by the system mask assigned to their user ID. This allows control over specific system requests to be limited to particular users if so required.

Chapter 5: Implementing Activity Logs

This section contains the following topics:

Activity Logs

The activity logging facility records all the activity in your region. You can use the activity logs to help determine the cause of problems.

Two separate activity log formats exist:

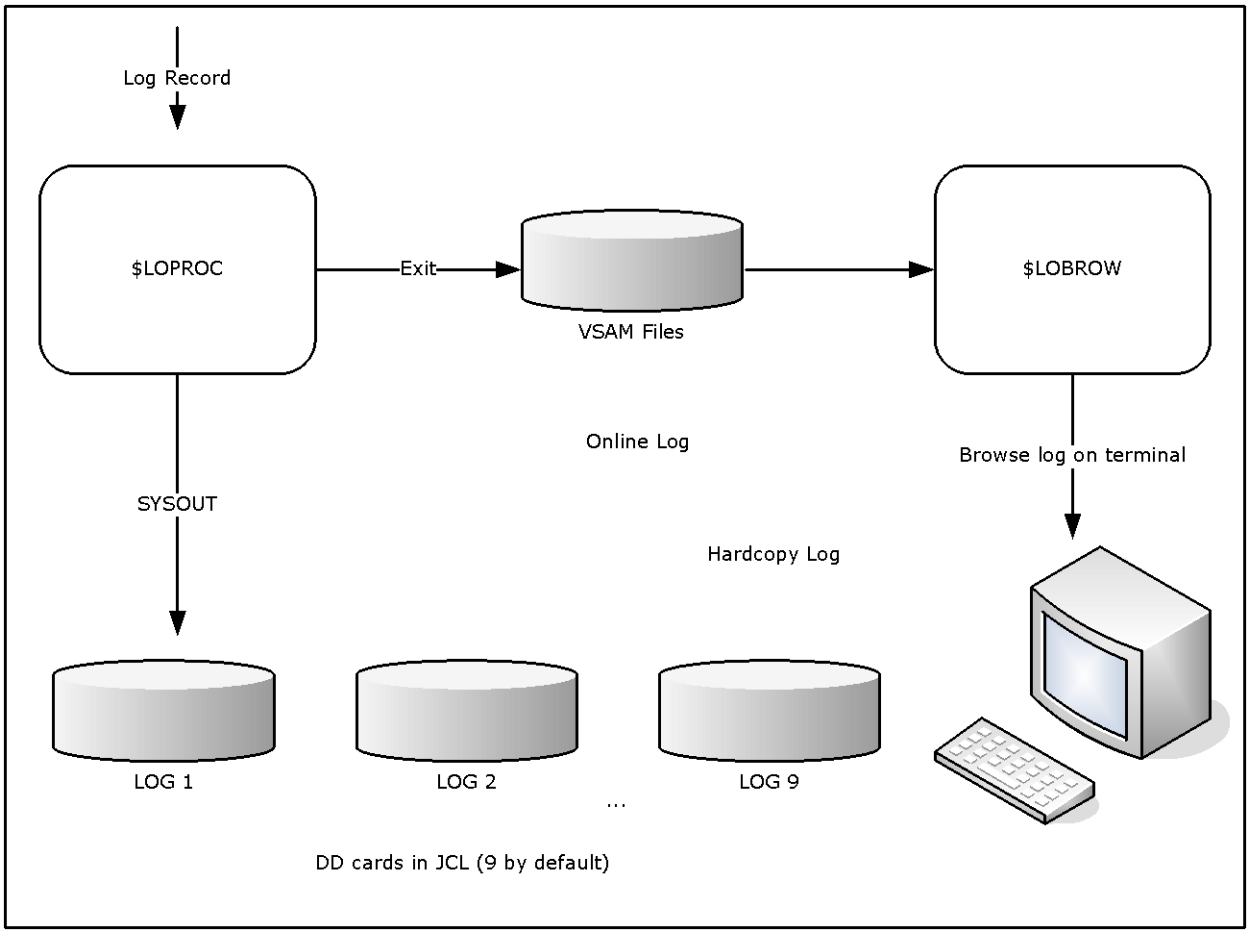
- Online
- Hardcopy

Log records are written to both formats.

By default, activity logs contain the following information:

- All commands entered
- All responses to commands entered
- Any unsolicited messages received from VTAM or the operating system, provided the related interfaces are available
- All messages explicitly written to the log by NCL procedures

The following illustration shows the path that the log record takes in the system.



The online activity log is supplied by the distributed procedure $LOPROC. The $LOPROC procedure writes log data to VSAM files (three by default). The VSAM files are accessed by a second procedure, $LOBROW, which allows online browsing of the log.

Note: $LOPROC and $LOBROW are the default procedure names. You can change these names by using the LOGFILES parameter group in Customizer (**/PARMS**).

Implement Online Activity Logging

During initialization, the region allocates, by default, three VSAM log files for online logging. However, you can allocate up to seven files.

Note: The log file IDs are of the form NMLOG*nn* and the data set names are of the form *dsnpref.rname*.NMLOG*nn*. (*dsnpref* is the data set prefix used during product installation and *rname* is the name of the region.)

Use Additional Log Files

If you want to make more than three files available to the region, define the new VSAM files and then customize the LOGFILES parameter group by defining additional logging data sets.

To customize the LOGFILES parameter group

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups list appears.

2. Enter **U** beside the LOGFILES parameter group.

   The Customizer : Parameter Group panel for the LOGFILES parameter group appears.

3. Press F8 (Forward) to display the next page.

4. Complete the fields for each file you want to make available. To allocate more files, press F8 (Forward) again.

5. When you have specified the required number of log files, press F6 (Action) to allocate and open the files.

6. Press F6 (Action).

   The changes are applied.

7. Press F3 (File).

   The changes are saved.

Note: For more information about using this panel, press F1 (Help).

Administer Online Activity Log Files

From the Activity Log : Administration menu, you can do the following:

- Swap active activity logs
- List all days contained in log files and browse logs for a particular date
- List all log files and browse a particular file

To administer online activity log files, enter **/LOADMIN** at the prompt.

The Activity Log : Administration menu appears.

Note: For information about the options available on this menu, press F1 (Help).

Swap the Online Log

The online activity log automatically swaps to a fresh VSAM file when each file fills up.

You can manually swap your currently active VSAM file if you want to free a particular log file (for example, for backups).

Important! Swapping the current VSAM log causes the $LOPROC procedure to write all subsequent activity log records to the next VSAM log. If this log was previously used, it is reset. Therefore, you can no longer browse the old records that it contained.

To swap the online activity log

1. Enter **/LOGSWAP** at the prompt.

   The Activity Log Services : Confirm Swap Log panel appears.

2. Press F6 to request the log swap, or F12 to cancel your request.

   Note: If the $LOPROC procedure encounters a VSAM error when it is logging activity to an online log file, it automatically swaps to the next log file.

Online Log Exit

You can create an NCL procedure to intercept, analyze, and react to the messages that are sent to the activity log.

Use the LOGFILES parameter group in Customizer to specify the name of your exit.

The exit is executed every time a message is sent to the log. Using the exit to perform complex functions can degrade the performance of the region.

Note: Ensure that your log exit procedure is well-tested before you put it into production.

Variables Available to the Activity Log Exit

The following variables are available to the activity log exit:

&#LO$RECORD

Contains records of the following formats:

time_generated user_id terminal_id message_text

The text of the message starts at the fourth word of the record.

arrival_time origin region* $$AOMTIME$$ *aom_time message_text

The text of the message starts at the sixth word of the record. This format lets you identify AOM-sourced messages.

You can change the contents of this variable. To suppress the message from the log, set the variable to NOLOG.

Note: For more information, see the &LOGREAD verb in the *Network Control Language Reference Guide*.

$LOG

Specifies a Mapped Data Object (MDO) that contains the message attributes. The MDO is mapped by the $MSG map.

You can use the &ASSIGN verb to query the MDO.

Note: For information about querying MDO components and additional variables, see the *Network Control Language Programming Guide*.

Example: Remove Messages from the NCL Log

The following shows an example procedure:

```
&CONTROL
-*-----------------------------------------------*
-* TO REMOVE IKJ56247I MESSAGES FROM THE NCL LOG. *
-*-----------------------------------------------*
&PARSE DELIM=' ' VARS=#LO$WORD* DATA=&#LO$RECORD
&IF .&#LO$WORD4 EQ .IKJ56247I &THEN +
        &#LO$RECORD = NOLOG
```

Enable the Log Exit

To enable the log exit

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups list appears.

2. Enter **U** beside the LOGFILES parameter group.

   The Customizer : Parameter Group panel for the LOGFILES parameter group appears.

3. Enter the name of your activity log exit in the Log Exit Name field.

4. Press F6 (Action).

   The changes are applied.

5. Press F3 (File).

   The changes are saved.

Online Logging Procedure

The default online logging procedure is $LOPROC. This procedure is designed to work with the online browsing procedure $LOBROW.

You can replace the $LOPROC and $LOBROW procedures with your own customized NCL procedures. Alternatively, you can write a customized log browsing procedure to present the supplied data files (from $LOPROC) in your own format.

Structure of Supplied Log Files

The supplied log files (NMLOG01, NMLOG02, and NMLOG03) have the following physical file structure:

- The record key has the following format:

  YYYYMMDDHHMMSSHS*nnnn*

  nnnn=1000 + (reset every 100th of a second) and key length=20 bytes

- A record has the following contents

  ORIGIN

  Contains the terminal name.

  REGION

  Contains the user ID.

TEXT

Contains the message text to display in the activity log.

MSGATTR

Contains the 2-byte color/highlight indicator. Colors are R=red, Y=yellow, W=white, B=blue, G=green, T=turquoise, or P=pink. Highlight values are R=reverse, B=blink, U=underscore, or N=none.

ORIGTIM

Contains the time at the remote domain.

ORIGDMN

Contains the name of the originating domain.

ORIGSRC

Contains the ID of the remote terminal.

Note: For more information, see the following references:

- The description of the &FILE OPEN verb in the *Network Control Language Reference Guide*.
- The *Network Control Language Programming Guide*.

How You Write Logging and Browsing Procedures

To write your own customized NCL procedure to replace $LOBROW, use the &FILE OPEN statement with FORMAT=DELIMITED.

You can store your log records in whatever file format you want. Your log browsing procedure must match this file format.

Note: For more information, see the descriptions of the following verbs in the *Network Control Language Reference Guide*:

- &LOGREAD
- &LOGCONT
- &LOGDEL

Implement Logging and Browsing Procedures

After you write your own browsing procedure or your own logging and browsing procedures, you implement them for use.

To implement your procedures

1. Enter **U** next to the LOGFILES parameter group in Customizer.
2. Update the relevant fields.
3. Press F6 (Action).

   Your procedures are used for logging and browsing.
4. Press F3 (File).

   Your changes to the parameter group are saved.

Hardcopy Activity Log

A region can have more than one hardcopy activity log, of which only one is open for logging.

Your region can be configured to perform logging to disk, tape, or hard copy. From one to nine logs can be specified by including the required number of DD statements in the execution JCL. Logging can be specified to wrap when the last log is full or is swapped.

To obtain the status of these logs, use the SHOW LOGS command.

Note: When logging to disk the following DCB attributes should be used:

```
DSORG=PS,RECFM=VBA,LRECL=137,BLKSIZE=15476
```

Format of Logged Information

Each entry recorded on the log has the following format:

```
12.04.23.12   SMITH      TERM54     +V NET,ACT,ID=NCP001
```

This entry consists of the following information:

- A time stamp in the format *hh.mm.ss.hs* (where *hh* is the hour, *mm* is the minute, *ss* is the second, and *hs* is the hundredth of a second)
- The user ID that entered the command or logged the message
- The terminal from which the command was entered or to which a message is sent
- The text of the message or command

Commands are highlighted with a plus sign (+) prefixed to the text to make it easier to distinguish commands from messages when browsing the log. If the command entered is an unsolicited VTAM command, it is highlighted and prefixed with an equals sign (=).

Format of Logged Timer-initiated Commands

Commands executed as the result of a timer-initiated command are prefixed by a plus sign, followed by the identity number of the timer command responsible. This identity number has the following format: #*nnnn*.

Example: Logged Timer-initiated Command

This example shows the log record of a command initiated by a timer:

```
15.00.00.01   NETOPER   CNTL01    +#0005 D BFRUSE
```

Format of Logged Commands Executed in Background Environments

Commands executed under the control of background environments are identified by the following keywords in the user ID field for the command text and any resulting messages:

BG-SYS

Background System Processor

BG-MON

Background Monitor

BG-LOG

Background Logger

Format of Logged Commands from NCL Procedure-dependent Environment

If a command is executed from an NCL procedure-dependent environment (&INTCMD), the node field on the log contains the NCL ID of the process issuing the command.

Format of the Hardcopy Log

The hardcopy log data set has the following format:

- A heading on each page—contains the day and date on which the log was created and the system identifier (NMID) of the originating region.
- A log identifier on the right side of the page. The log identifier is the ddname under which the log was created. This log identifier assists log collation after printing.
- 60 lines on each page—this format can be altered to suit your requirements using the SYSPARMS LOGPAGE operand.

  Note: For information about LOGPAGE, see the *Reference Guide*.

Swap the Hardcopy Log

Swapping the current log frees the log for immediate printing. Swapping the log is possible only when another unused log remains to which logging can continue. You can specify up to nine logs. Logs do not need to be consecutive.

To swap the log, enter the LOGSWAP command.

When a log is swapped, the log status, the requesting user ID, and the reason for the swap are recorded. You can display these details with the SHOW LOGS command.

Each of the logs is identified in the JCL member by the LOG*n* ddname. *n* is in the range one to nine.

Example: Log Name

This example defines the LOG4 ddname:

```
//LOG4 DD SYSOUT=A,FREE=CLOSE
```

Mixing of device types is valid. Inclusion of FREE=CLOSE prints the log when it is released by the LOGSWAP command.

Reuse of Hardcopy Log Data Sets

Wrapping lets you reuse a LOG data set when all of the available LOG data sets have been used.

The LOGWRAP SYSPARM determines whether log data set wrapping is allowed. You set the value of this SYSPARM in the Are Activity Logs to Wrap? field when you customize the LOGFILES parameter group in Customizer (**/PARMS**).

If you specify NO (the default) in the Are Activity Logs to Wrap? field, then wrapping is not permitted. When all the LOG data sets have been used due to successive LOGSWAP commands, the previous LOG data sets cannot be reused. After the last LOG data set is used, any further LOGSWAP commands are rejected.

If you specify YES in the Are Activity Logs to Wrap? field, log wrapping is allowed according to the following rules:

- If you direct your LOG data sets to SYSOUT, then, as each LOG*n* DD statement is used, the data set is unallocated because FREE=CLOSE. In this case, you can reissue an ALLOC command to reallocate another SYSOUT file under the same ddname. For example:

  ```
  ALLOC DD=LOG3 SYSOUT=A FREE=CLOSE
  ```

  This ddname is now available for use as another LOG data set. Subsequent LOGSWAP operations can now reuse this LOG data set rather than rejecting the command when the last LOG data set is used.
- If the LOG DD statements point to sequential data sets, log wrapping overwrites the earlier LOG data held in these data sets. Archive the existing data before allowing the wrap to occur.

Cross-Reference of Hardcopy Logs

To help operations staff to piece the full log together, certain information is recorded on the last and first lines of swapped LOG data sets.

The first line of a new log contains the reason for the swap, or the initiating user ID.

The last message printed on a swapped log is the ddname of the new log. Also printed at the start of the new log is the ddname or logical ID for the previous log.

I/O Errors on the Hardcopy Log

If an I/O error occurs on a log, the log is closed and the next available log is automatically swapped to, if one is available, and logging continues. This also applies to data set full conditions when logging to disk.

If the I/O error occurs on the last available log, a warning message is sent to all monitor terminals informing them that logging has ceased. The STATUS command also includes a warning message if logging is stopped. All log messages are passed to LOGPROC for analysis even if no log output is possible.

Write to the System Log

You can use the SYSPARMS SYSLOG operand to write all logged output or all VTAM PPO messages received to the system log.

To write all logged output to the system log also, enter the **SYSPARMS SYSLOG=YES** command.

To write all VTAM PPO messages to the system log also, enter the **SYSPARMS SYSLOG=PPO** command.

Note: For more information about the SYSPARMS SYSLOG operand, see the *Reference Guide*.

Chapter 6: Setting Up the Initialization File

This section contains the following topics:

Generate an Initialization File

If you are deploying multiple regions, each region must be configured for its local environment. When you have configured your first region, you can build an initialization file from that region and then configure it for use with your other regions. This removes the need to customize each region with Customizer.

The tasks outlined below show how to configure a region from an initialization file. The initialization file is produced from a running region for your product.

To generate an initialization file

1. From the Primary Menu, enter **/CUSTOM**.

   The Customizer panel appears.

2. Select option G - Generate INI Procedure.

   The Customizer : Generate INI Procedure panel appears.

3. Enter the data set name and the member name of the file in the Generate INI File Details section.

   Note: The data set must be in the commands concatenation of the RUNSYSIN member for the region in which it is used.

4. Ensure that the member name and data set name are correct. Enter **YES** in the Replace Member? field if you are replacing an existing member.

5. Press F6 (Action).

   The initialization file is generated.

6. Make a note of the data set and member names and press F6 (Confirm).

   The details are saved.

How You Configure the Initialization File

The initialization file must be configured before it can be used for other regions. You can perform this configuration as follows:

- Configure an individual initialization file for each region.
- Configure a common initialization file for multiple regions.

You can use system variables and static system variables with both of these methods. The variables substitute for the initialization parameters in the INI file.

Configure a Common Initialization File

You can customize an initialization file using variables so that it can be used for multiple regions.

To configure a common initialization file

1. Create a data set that is available to every region to be initialized from the common initialization file, for example, PROD.INIFILES.
2. Add the newly created data set to the COMMANDS concatenation of the RUNSYSIN member to every region to be initialized from the common initialization file.

   Note: RUNSYSIN is located in TESTEXEC.
3. Copy the initialization file generated into the new INIFILES data set.
4. Use your TSO editing tool to open the initialization file in edit mode.
5. Replace the relevant generated variables in the initialization file with the following system variables:

   &ZDSNQLCL

   The local VSAM data set qualifier.

   &ZDSNQSHR

   The shared VSAM data set qualifier.

   &ZACBNAME

   The primary VTAM ACB name used by the region.

   &ZDSNQLNV

   The local non-VSAM data set qualifier.

   &ZDSNQSNV

   The shared non-VSAM data set qualifier.

   &ZNMDID

   The domain identifier.

&ZNMSUP

The system user prefix.

6. Replace the relevant generated variables in the initialization file with the z/OS static system symbols as follows:

   &SYSCLONE

   The short name for the system.

   &SYSNAME

   The name of the system.

   &SYSPLEX

   The name of the sysplex.

   &SYSR1

   The IPL VOLSER.

7. Save the changes to the initialization file.

Configure Individual Initialization Files

You can customize an initialization file generated from one region so that it can be used for another region.

To configure an individual initialization file for each region

1. Use your TSO editing tool to open the initialization file in edit mode.
2. Substitute the parameters in the initialization file with *one* of the following:
   - Hard-coded data set names for the region in which the file is used
   - System variables

   This enables the initialization file to work in regions with different data sets than the region in which it was generated.
3. Save the changes to the initialization file.
4. Copy the initialization file to the region's TESTEXEC or one of the other libraries in the COMMANDS concatenation.
5. Repeat steps 1 to 4 for each initialization file needed.

Note: The region from which the original initialization file was generated should have the same product sets as the destination regions that will use that initialization file.

Start Your Region from an Initialization File

The name of the initialization file must be specified by the INIFILE parameter in the RUNSYSIN member.

Updating your RUNSYSIN member causes your region to set its initialization parameters from the initialization file. All Customizer parameter settings are overwritten.

To update your RUNSYSIN member

1. Use a text editor to open your RUNSYSIN member.
2. Insert the line PPREF='INIFILE=*membername*' into your RUNSYSIN member.
3. Save the member.

Chapter 7: Implementing Print Services

This section contains the following topics:

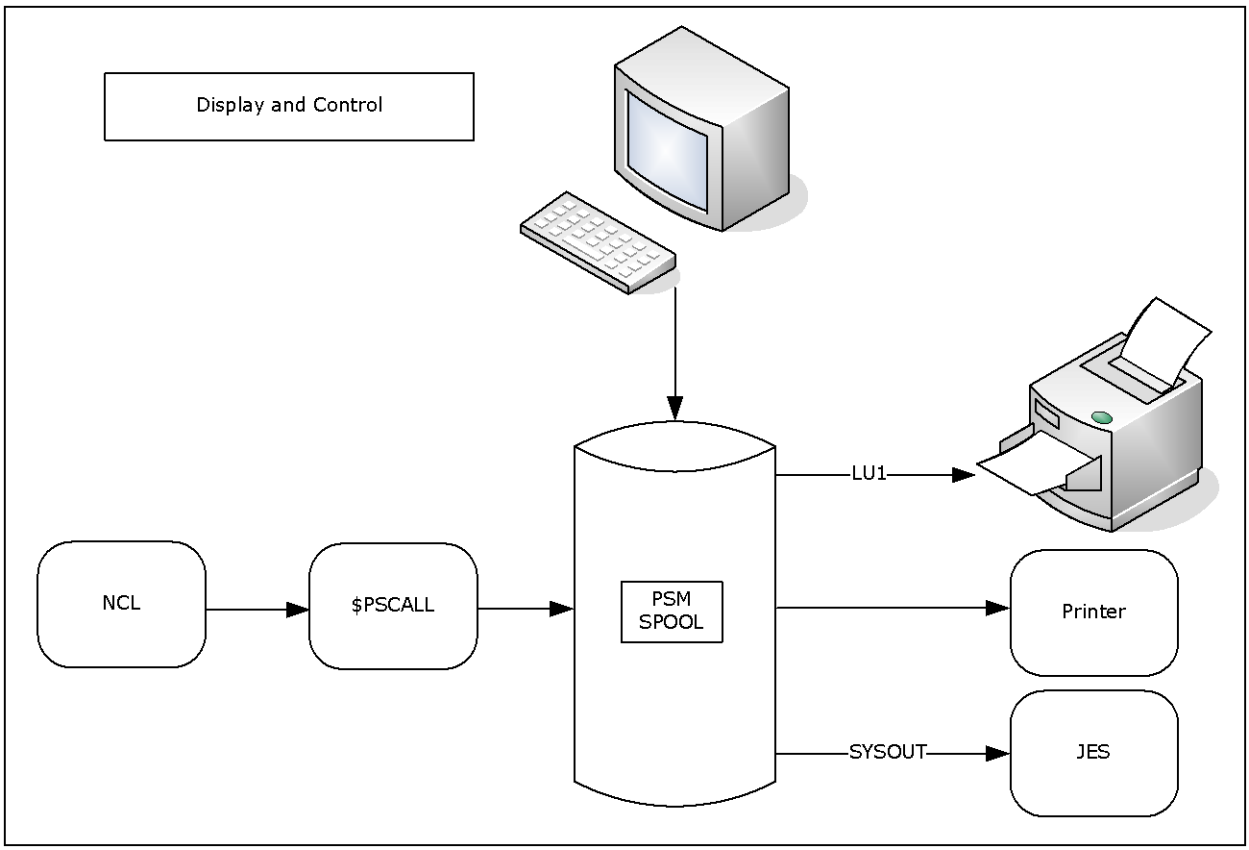
Print Services Manager

Print Services Manager (PSM) allows you to specify the format of a print request and on which printer it is printed. Print requests can be viewed online before or after printing and can be redirected to files rather than printers.

PSM provides the following features, which can be customized to suit your requirements:

- Printer definition facilities
- Form definition maintenance
- Setup definition maintenance
- Default printer assignment maintenance
- Alias printer name definitions
- Banner page customization on output
- Spooled print request browsing, retention, and redirection to a different printer
- Integration with NCL-based components

The following illustration shows the different ways that PSM can be used to control printing requirements.



Access PSM

The customizable functions of PSM are accessed from the PSM : Primary Menu.

To access PSM, enter **/PSM** at the prompt.

Note: You can also access PSM directly by invoking the $PSCALL NCL procedure from OCS or an installation written NCL procedure. The PSM NCL interface is described in the *Network Control Language Reference Guide*.

Add a Printer Definition

A printer definition defines where, how, and on what paper output is printed. A printer definition is required for each printer at which output is printed.

To add a printer definition

1. Enter **/PSMPRTR** at the prompt.

   The PSM : Printer Definition List appears.

2. Press F4 (Add).

   The PSM : Printer Definition panel appears.

3. Complete the fields, as required.

   Note: For information about the fields, press F1 (Help).

4. Press F3 (File).

   The definition is saved.

List Printer Definitions

You can display a list of all the printer definitions defined for your region. This lets you browse and perform maintenance on the listed definitions.

To list all printer definitions, enter **/PSMPRTR** at the prompt.

Add a Form Definition

A form definition is required for each type of paper on which output is printed. The Form Definition Menu is used to set up and administer these form definitions.

To add a form definition

1. Enter **/PSMFORM** at the prompt.

   The PSM : Form Definition List appears.

2. Press F4 (Add).

   The PSM : Form Definition panel appears.

3. Complete the fields and press F3 (File).

   The form definition is saved.

   Note: For information about the fields, press F1 (Help).

List Form Definitions

You can list all of the form definitions defined for your region and then browse and perform maintenance on them.

To list all form definitions, enter **/PSMFORM** at the prompt.

Add Control Characters

Control characters are sent to a printer before or after (or both) the output is printed. They are defined in setup definitions.

To add control characters

1. Enter **/PSMSET** at the prompt.

   The PSM : Setup Definition List appears.

2. Press F4 (Add).

   The PSM : Setup Definition panel appears. To access the second panel of the setup definition, press F8 (Forward).

   Complete the fields, as required.

   Note: For information about the fields, press F1 (Help).

3. Press F3 (File).

   The setup definition is saved.

List Control Characters

You can display a list of all the setup definitions defined for your region. This list lets you browse and perform maintenance on the listed definitions.

To list control characters, enter **/PSMSET** at the prompt.

Add a Default Printer for a User ID

Each user ID in your region can be assigned a default printer. Default printer assignments let you define the printer to which output is sent whenever a user ID does not specify a printer.

To add a default printer for a user ID

1. Enter **/PSMDFTP** at the prompt.

   The PSM : Default Printer Assignment List appears.

2. Press F4 (Add).

   The PSM : Default Printer Assignment panel appears.

3. Complete the following fields:

   User ID

   Specifies the User ID of the user to whom the printer is assigned a default.

   Printer Name

   Specifies the name of the printer to which this user's printing is sent.

   Press F3 (File).

   The default printer assignment is saved.

List Default Printers

You can display a list of all the default printer assignments defined for each user ID. This list lets you browse and perform maintenance on the listed definitions.

To list default printers, enter **/PSMDFTP** at the prompt.

Clear the Printer Spool

Print requests are retained on the print spool if an error occurs during printing or if HELD is specified on the PSM : Print Request panel. The PSM clear spool panel is used to clear print requests from the print queue.

Note: This function is available to authorized users only.

To clear the print spool

1. Enter **/PSMADMN** at the prompt.

   The PSM : Administration Menu appears.

2. Enter **CS** at the prompt.

   The PSM : Clear Spool panel appears.

3. Complete the following field:

   Date

   Specifies that all print requests added to the spool before or on this date are deleted.

   Press F6 (Action).

   The print requests are deleted.

Exits to Send Print Requests to a Data Set

Two printer exit procedures are distributed with your product. Each writes the output for a print request to a data set. The procedure $PSDS81X can be customized to specific site requirements. The procedure $PSDS81Z offers the same functionality with improved performance, but cannot be customized. The target data sets for both procedures can be sequential or partitioned.

Parameters that control the operation of the exit are defined in the Exit Data portion of the printer definition. Procedures that pass data to PSM for printing can override the exit data specified in the PSM printer definition.

The procedures use the parameters contained in the exit data to do the following:

- Determine the target data set
- Determine how to process a data line with a skip amount of zero
- Set the length of the lines print

How the Procedures Process a Print Request

The procedures read each line of print data and write it directly to the nominated data set. Each print line is analyzed according to skip control before processing. This continues until all lines of data for the print request have been received from PSM and written to the nominated data set.

$PSDS81X and $PSDS81Z Parameters

The $PSDS81X and $PSDS81Z exits have the following keyword parameters:

```
  DSN=datasetname
[ DISP={ SHR | OLD | NEW | MOD } ]
[ LRECL={ n | 80 } ]
[ SKIP0={ NEWLINE | DISCARD | DESTRUCTIVE |
          NONDESTRUCTIVE } ]
[ CYL= pri [,sec] [,dir] ]
[ TRK={ pri [,sec] [,dir] | 15,5 } ]
[ BLKSZ= n ]
[ STORC= storclas ]
[ MGMTC= mgmtclas ]
[ DATAC= dataclas ]
[ VOL= volser ]
[ UNIT={ unit | SYSALLDA} ]
[ RECFM={ F | FB | V | VB } ]
```

DSN=*datasetname*

Specifies the target data set name. If the data set is partitioned, the member name must be included or the data set is corrupted.

You can use the following symbolics in the *datasetname* parameter:

- &DAY is the day of the week (for example, MON).
- &YY is the two-digit representation of the year (for example, 11).
- &YYYY is the four-digit representation of the year (for example, 2011).
- &MM is the two-digit representation of the month (for example, 02).
- &MON is the three-character representation of the month (for example, JAN and FEB).
- &DD is the day of the month.
- &HHMMSS is the time.
- &HH is the hour.
- &MIN is the minute.
- &JOBID is the job ID.
- &JOBNAME is the job name.
- &NMID is the region ID.
- &NMDID is the region domain ID (DID).
- &GRPNAME is the sysplex name.
- &SYSID is the system ID.
- &SYSNAME is the system name.
- &USERID is the requesting user ID.

Symbolics are delimited by a period (.) or another symbolic (that is, &YY&MM. is the same as &YY.&MM.). Symbolics are also allowed in a member name.

Example:

```
DSN=NM.&SYSID..&USERID..D&YY&MM&DD..T&HHMMSS..DATA
```

For example, this specification can resolve to the following data set name:

```
DSN=NM.SYSA.MYUSER.D040915.T144505.DATA
```

DISP={ SHR | OLD | NEW | MOD }

Specifies the disposition of the output data set.

- SHR specifies shared use of the data set.
- OLD specifies exclusive use of the data set.
- NEW allocates a new data set.
- MOD appends the output in the file.

Default: SHR

LRECL={ *n* | 80 }

Specifies the output record length.

Limits: 1 through 250

Default: 80

SKIP0={ NEWLINE | DISCARD | DESTRUCTIVE | NONDESTRUCTIVE }

Specifies how to process a data line with a skip amount of zero.

- NEWLINE creates a line of data.
- DISCARD discards the line of data.
- DESTRUCTIVE causes the data to replace the existing data line.
- NONDESTRUCTIVE overlays the data on the existing data line, but only where blanks were present on the existing data line. No existing non-blank characters are modified.

Note: The procedures ignore the following PSM print options: NEWPAGE and USCORE.

Default: NEWLINE

The following additional parameters are applicable when DISP=NEW is specified:

CYL=*pri,sec,dir*

Specifies the primary and secondary space allocation values in cylinders. If a partitioned data set is used, the parameter specifies the number of directory blocks.

TRK=*pri,sec,dir*

Specifies the primary and secondary space allocation values in tracks. If a partitioned data set is used, the parameter specifies the number of directory blocks.

Default: TRK=15,5

BLKSZ=*blocksize*

Specifies the block size.

STORC=*storclas*

Specifies the storage class.

MGMTC=*mgmtclas*

Specifies the management class.

DATAC=*dataclas*

Specifies the data class.

VOL=*volser*

Specifies the volume serial number.

UNIT= { *unit* | SYSALLDA }

Specifies the unit.

Default: SYSALLDA if volser is specified

RECFM= { F | FB | V | VB }

Specifies the record format.

Default: FB

Printer Exit Definition Example

This example directs the output for a PSM print request, assigned to the printer named DSEXIT, to the member TEST1 in the data set PROD.PSM.DATA. The record length of this data set is 80. Overlay lines in the data are removed.

```
PROD1--------------------- PSM : Printer Definition --------------------------
Command ===>                                                 Function=BROWSE

  Printer Name ... DSEXIT
  Type ........... EXIT              (JES, VTAM, ALIAS, EXIT)
  Description .... Print to a data set
  Lower Case? .... YES               (Yes or No)
  Line Limit ..... 0                 (0 to 999999)
  Form Name .....+ FORM0
ALIAS Printer
  Real Name .....+                   (Real printer name)
JES Printer
  Destination ....                   (destid.userid)
  Output Class ...                   (A to Z, 0 to 9)
VTAM Printer
  LU Name ........
  Logmode ........
EXIT
  Exit Name ...... $PSDS81Z
  Exit Data ...... DSN=PROD.PSM.DATA(TEST1) LRECL=80
                   SKIP0=DISCARD
```

Note: Previous references to parameters WKVOL, CYL, and LIST in the exit data are no longer required. Remove them from the printer definition before using $PSDS81Z or $PSDS81X, or the print request fails.

Print-to-Email

The $PSEMAIL printer definition lets you email the output of a printing request. The request can be either an attachment or in the body of the email. When the output is sent as an attachment, the email uses the PS8803 message as its body and the PS8804 message as its salutation:

`Data attached for` *`email_subject`*

`Yours,`
`user_name`

user_name

Displays the sender name defined in UAMS.

You can maintain these messages from the Message Definition List panel. The shortcut to the panel is /CASMSG.

Note: For information about how to maintain messages, see the *Managed Object Development Services Guide*.

Chapter 8: Controlling Issued Commands

This section contains the following topics:

Overview

System transmission definitions may include a command to execute when the transmission terminates, either at the transmitting end, the receiving end, or both. This facility is typically used to send a message or submit a job for execution that depends on the receipt of the data.

Commands issued on completion of a transmission are executed in the background system (BSYS) environment. To restrict the commands that can be issued, you can adjust the authorization level of BSYS. However, because BSYS is used by other system components, this is not always desirable. A more flexible means of restricting these commands is by implementing FTSCPROC.

FTSCPROC

FTSCPROC is an NCL procedure that is executed when a terminating transmission attempts to issue a command. The same NCL procedure is invoked for incoming and outgoing transmissions.

When FTSCPROC is invoked, it is passed information about the terminating transmission and the command to execute. This information is passed in seven parameters and is available in NCL variables &1 to &7.

The contents of these variables differs slightly for incoming and outgoing transmissions:

- For incoming transmissions:

  &1

  The command (maximum 40 characters)

  &2

  Set to R to show that the transmission has just been received *(incoming)*

  &3

  The remote requestor's user ID

  &4

  The name of the transmission

  &5

  Type of the transmission, system (S) or private (P)

  &6

  The name of the originating system

  &7

  The originating system type (H for host)

- For outgoing transmissions:

  &1

  The command (maximum 40 characters)

  &2

  Set to T to show that the terminating transmission was just transmitted (outgoing)

  &3

  The requestor's user ID

  &4

  The name of the transmission

  &5

  Type of the transmission, system (S) or private (P)

  &6

  The name of the destination system

  &7

  The destination system type (H for host)

Using this information, your FTSCPROC can issue the command, issue a different command, or suppress the command.

FTSCPROC gives you flexibility in controlling commands issued by CA SOLVE:FTS. An example of a possible implementation is ensuring that transmissions from a particular host (perhaps one that is not part of your organization) can issue only a restricted set of commands.

Note: A sample FTSCPROC, named $FTCPROC, is included in the installation library.

Activate and Deactivate FTSCPROC

Activating and deactivating FTSCPROC is controlled by SYSPARM FTSCPROC.

To activate FTSCPROC, enter:

```
SYSPARM FTSCPROC=procname
```

To deactivate FTSCPROC, enter:

```
SYSPARM FTSCPROC=NONE
```

Chapter 9: Implementing the NCL Interface

This section contains the following topics:

NCL Procedure Descriptions

A number of NCL procedures are supplied to form the standard NCL Interface. A description of each procedure follows, together with details of calls made to other procedures.

NCL Access to the VFS

NCL procedures have access to VSAM data sets (termed user databases, or UDBs) by using &FILE statements. A UDB can be in internal format or external format. External format UDBs are those usually created by a means other than NCL procedures, but NCL procedures can read and update these data sets.

To access a UDB, an NCL procedure must establish a logical connection to the data set using the &FILE OPEN verb, where the ID= operand identifies the logical file name that has been previously assigned through a UDBCTL command. In the case of the transmission definitions, the logical file ID is VFS.

Records contain nondisplay (hexadecimal) characters. These records must be processed in expanded hexadecimal format to preserve data transparency. For example, record keys are padded with binary zeros (X'00') rather than blanks (X'40'). Unless the key and data fields are converted to expanded hexadecimal format, NCL converts nondisplay characters such as binary zeros. For this reason, we recommend that you access all transmission definition records by using the $FTSDGET, $FTSDPUT, and $FTSDDEL NCL procedures.

Important! Extreme care must be taken if the supplied procedures are modified. Invalid records written to VFS could severely impact the execution of CA SOLVE:FTS and cause an abend. Records, other than those manipulated by the supplied procedures, should not be modified.

Note: For information about NCL access to UDBs in *external format,* see the *Network Control Language Programmer Guide.*

Access to File Definition Facilities

The following NCL procedures provide a low-level interface to the Transmission Definition facilities:

$FTSDEMO

Provides an example of the type of tailored display available and the way to handle on demand requests. The procedure requests that you enter only a name for the transmission (which must start with your user ID, and this user ID must have private authority), the destination, the transmission class, and the input and output file names and dispositions. This information is then edited, and defaults are taken for all other options. The procedure then does the following:

1. Calls $FTSDPUT to write a transmission definition record to VFS as a private definition specifying conditional definition deletion.
2. Executes an XMIT command to request transmission and waits for a response.
3. Returns the response to the XMIT command to your panel.

$FTSDDEL

Deletes a transmission definition record from VFS.

$FTSDGET

Retrieves a transmission definition record from VFS and makes the options available in the &FTS variables.

$FTSDPUT

Writes a transmission definition record to VFS, using the options available in the &FTS variables.

$FTSDSFR

Prompts you with standard From panel and primes the &FTS variables with the information obtained from it.

$FTSDSSO

Prompts user with standard SYSOUT panel and primes the &FTS variables with the information obtained from it.

$FTSDSTO

Prompts user with standard To panel and primes the &FTS variables with the information obtained from it.

Limitations of Procedures

The following limitations apply to the implementation of the procedures:

- Data set passwords are not supported.
- The editing implemented in the NCL procedures is not as comprehensive as that provided by the base CA SOLVE:FTS. Depending on the functions utilized, this can lead to errors at transmission time that currently may be picked up at definition time. For example, if data set names are not edited comprehensively, dynamic allocation failures may occur.
- The target data set DCB information is not edited comprehensively—there is no cross-check between LRECL, BLKSIZE and RECFM to ensure consistency. This may cause data set OPEN failures.
- No cross-checks are made between partitioned data set allocation and the provision of a member name (with the exception of the ISPF Dialog components). This may cause data set OPEN failures.

Name Masking

The $FTSDEMO procedure implements name masking so that a user can only request a transmission whose name begins with their user ID.

If you want to eliminate this masking technique, remove the &MASK=&USERID statement at the beginning of those procedures or set the mask to an alternative value from 1 to 12 characters using a standard assignment statement.

Generate Transmission Definitions Dynamically

The $FTSDEMO procedure generates a transmission definition dynamically, based partly on user input and partly on prescribed defaults. The new definition is added specifying conditional definition deletion and a TRANSMIT command issued to request the transmission. If the transmission completes successfully, the definition is deleted automatically; otherwise, it remains and may be amended and requested again in the usual manner. Alternatively, procedure $FTSDEMO may be modified so that definition deletion occurs regardless of the transmission's success (by setting &FTSDELDF to Y). In this case, the same steps used to originally define the request may be required to recover from failed transmissions.

These techniques can be utilized in many ways. The information necessary to create a transmission definition can be obtained from a variety of sources, such as a UDB, from full-screen panels, or from parameters passed to an NCL procedure. For example, an NCL procedure based on $FTSDEMO can be passed only the names of the input and output data sets to set defaults for all other options before requesting the transmission.

Such a procedure can also be invoked from a batch job through UTIL0001 (see page 17). The step of the batch job that executes UTIL0001 could submit the following command:

```
F NM,EXEC FTSSUB INPUT.DATASET OUTPUT.DATASET
```

This has the effect of executing the NCL procedure FTSSUB under control of the SYSOPER user ID, passing it the names of the input and output data sets. This procedure then defines a transmission and requests it.

Access to $FTCALL API

The following procedures provide entry to the Application Program Interface procedure, $FTCALL:

$FTSADD

Invokes FTS Transmission Definition Maintenance to add a new transmission definition. You are prompted with the To and From details panels and the new definition filed.

$FTSDSPL

Invokes FTS Transmission Definition Maintenance to update an existing transmission definition. You are prompted with the To and From details panels and the new definition filed.

$FTSMENU

Displays FTS Primary Menu. This is the same as the **/FTS** shortcut.

$FTSMNT

Displays the FTS Transmission Definition Menu.

$FTSSEL

Lists FTS Transmission Definitions.

$FTSSTAT

Displays the FTS Transmission Supervision Menu or the Transmission Request List. For more information, see the comments in the procedure.

These procedures replicate some of the functions provided through the panels and are retained for compatibility with previous releases (when these functions were not available). We recommend that you access these functions through the FTS : Primary Menu.

Chapter 10: Implementing the ISPF Dialog Interface

This section contains the following topics:

ISPF Dialog Interface

The ISPF Dialog Interface integrates into the existing ISPF environment and provides a means of effecting file transmissions.

Although the dialog contains some additional functions, it does not implement all CA SOLVE:FTS options and is not intended as a replacement for the standard facility.

Any CA SOLVE:FTS region in the network that is accessible from the TSO host can perform file transmissions. The choice depends on the location of the source data set. If the source data set resides on the TSO host system, then the transfer is from the *local* host and is affected by the region running on the same CPU. If the source data set is on another host system, then it is considered a *remote* host and the transfer is affected by the region running on that remote CPU.

Note: Certain functions are restricted to local host transfers and to specific data set types.

Invoke the Dialog

Before you can use the ISPF Dialog for the first time, you need to include the ISPF components in your ISPF library concatenations. This process is site-specific so you may need to consult your site's TSO/ISPF administrator.

The following shows the ISPF data set type and data set name, and the associated CA SOLVE:FTS data set name:

CLISTs

ISPF data set name: SYSEXEC

CA SOLVE:FTS data set name: CDEMCLS0

Messages

ISPF data set name: ISPMLIB

CA SOLVE:FTS data set name: CDEMCLS0

Panels

ISPF data set name: ISPPLIB

CA SOLVE:FTS data set name: CDEMCLS0

To use the ISPF interface you must do *one* of the following:

- Amend your ISPF startup procedures to include the CA SOLVE:FTS data sets.
- Copy the contents of the CA SOLVE:FTS data sets into existing ISPF data sets for that data set type.

Note: The ISPF Dialog also uses the NMCMD TSO processor that was installed when CA SOLVE:FTS was installed.

The dialog is intended to run as an ISPF Dialog application. This means it has its own operational shared variable pool and its own profile data set member in which information is stored across application sessions.

The dialog application can be tested from Option 7.1 of ISPF (by invoking procedure NMFTS and specifying a NEWPOOL ID of NMFT for the call) but it should ultimately be invoked from some convenient user menu (for example, a tailored primary ISPF menu). The following dialog statement can be used to access the dialog system from such a menu:

```
ISPEXEC SELECT CMD(NMFTS) NEWAPPL(NMFT)
```

After it is invoked for the first time, the dialog application creates a member NMFTPROF in the user's ISPF profile data set on exit.

Transmission Request Names

Each transmission request in a given region must have a unique name. The dialog uses the TSO user's user ID plus one character (A to Z) in order to construct both a valid and unique transmission name. This lets the user have up to 26 requests outstanding in a given region. The dialog is responsible for determining the next character to use in constructing the transmission name. This name is returned in a message to the user after the request is successfully processed and the transmission queued.

Manage Temporary Data Sets

Dialog procedures are designed to eliminate temporary data sets, allocated for staged file transmissions, as soon as the request is complete. However, various errors are possible (such as link failure), which result in such data sets not being deleted by the usual system processes.

Since every user of the ISPF dialog is not aware of the various mechanisms involved in scheduling the transmission, it is assumed that the installation needs to perform some regular housekeeping procedures to delete any unwanted data sets.

This requires knowledge of the names allocated to temporary data sets. The installation has some control over these names by setting the variables &LCLPREF and &REMPREF through procedure NMFTSID (see page 89). This prefix is used to construct the temporary data set names used by the dialog in the following manner.

The following staging data set name is used in the sending system:

```
&LCLPREF.&FTSNM.NMSTGO
```

The following staging data set name is used in the receiving system:

```
&REMPREF.&FTSNM.NMSTGI
```

The following IEBCOPY control file name is allocated in the sending system:

```
&LCLPREF.&SYSUID.SYSIN
```

The variable &SYSUID is the TSO user ID of the person using the dialog and is a standard dialog system variable. The variable &FTSNM is the TSO user ID (&SYSUID) plus one character (A to Z), which is the transmission name constructed by the dialog to uniquely identify this request.

Only one IEBCOPY control file is required and is deleted after the output staging data set is created. The staging data set names are tied to the transmission request names for ease of reconstructing at the receiving end and also as a convenient means to make the name unique.

Depending upon the exact nature of the staged transmission, particular NCL procedures are invoked upon its successful completion in both the sending and receiving systems. In the sending system, the procedure $@DELETE is always invoked and the value of &LCLPREF, defined by the installation as the local staging data set prefix, is passed as a parameter to this procedure. It deletes the staging data set on the local system and notifies the TSO user of the successful transmission. In the receiving system, the prefix cannot be carried as a parameter and must be determined by calling the NCL procedure $@NMFTS. When determining what values to use for the &REMPREF variable defined in the dialog procedure NMFTSID, remember that procedure $@NMFTS, executing in the remote system, must be able to reconstruct such a prefix given only the original TSO user ID of the requestor. It returns this value in &PREFIX to the calling NCL procedure ($@COPY, $@REPRO, or $@E), which then constructs the entire staging data set name from which to reload the target data set and then delete.

Error Recovery

When problems do occur, it is not always possible to notify users that their file transmissions have failed. To determine the cause of the problem, you can log onto the source region and determine the cause of the transmission failure through standard panels. When the error is isolated, the request can be restarted if possible, or deleted and a new request issued. Remember that for staged transmissions, this may result in data sets not being deleted.

NCL procedures, executed after a successful file transmission, write messages to the activity log. This is a useful source of information.

More complete notification techniques, following successful or unsuccessful file transmissions, can be implemented by trapping the relevant messages (both standard messages and the messages issued by the NCL procedures executed at successful completion of the transmission) in an activity log exit procedure.

An installation can implement new NMCMD subcommands that let users track transmission requests and delete unwanted requests and data sets.

Dialog Components

The dialog comprises various ISPF Dialog Manager procedures, Assembler programs, panels, help panels, and message members, as well as supporting NCL procedures.

Dialog Procedures

NMFTS

The main entry procedure that controls the dialog flow. It calls procedure NMFTSID, which sets installation options and then presents the FROM and TO panels to collect information regarding the user's file transmission request. When all such information is available, it invokes one of three other procedures (NMFTSC, NMFTSM or NMFTSS, depending on the transfer option requested) to perform any actions related to the type of transfer.

NMFTSC

This is invoked when the C option (direct copy) is selected and all information regarding regions and data sets is available. If the source data set is a PDS on the local system and no member name was supplied, a full member selection list is supplied. The TSO command processor NMCMD is then invoked to schedule the transmission request with the appropriate region.

NMFTSM

This is invoked when the M option (direct copy specifying multiple members) is selected, and all information regarding regions and data sets is available. The source data set must be a PDS (verified if on the local system) and a panel is presented where you can enter up to 20 member names and new names, if they are to be renamed. The TSO command processor NMCMD is then invoked to schedule the transmission request with the appropriate region.

NMFTSS

This is invoked when the S option (staged copy) is selected and all information regarding regions and data sets is available. If the source data set is a PDS on the local system and no member name is supplied, a full member selection list is supplied. For VSAM files a panel allowing selected key ranges (applicable to KSDS data sets) is presented. After allocating the necessary staging data sets, the TSO command processor NMCMD is invoked to schedule the transmission request with the local region.

NMFTSID (see page 89)

This is an initialization procedure called from NMFTS when it is first invoked. It lets the installation specify various configuration options and defines the systems available for functions.

NMFTSED

This is called from NMFTS to edit the system names provided by the user for the functions. It verifies that the system is valid and that the target system is accessible from the source system, based on the information supplied in procedure NMFTSID.

Dialog Programs

NMFTDSRG

This is an Assembler program that is called to determine the data set organization of a requested data set. It expects the dialog variable &DSN1 to be in the shared variable pool and to contain the name (qualified or unqualified) of the data set in question. It sets a return code as follows:

0

Indicates that the function was successful.

8

Indicates that the data set was not cataloged.

For a return code of zero, NMFTDSRG sets the following dialog variables in the shared variable pool:

&ZDSORG

Defines the data set organization as *one* of the following:

- PO - Partitioned sequential
- PS - Physical organization
- VS - VSAM organization

&VOL1

Contains the actual volume on which the data set resides.

&ZRECFM

Contains the record format of the data set.

&ZLRECL

Contains the logical record length for the data set.

&ZBLKSZ

Contains the data set block size.

&ZUNITS

Contains the data set allocation units as CYLINDERS, TRACKS, or BLOCKS.

&ZPRISPC

Contains the primary space allocation quantity.

&ZSECSPC

Contains the secondary space allocation quantity.

NMFTMSEL

This is a dialog program that uses ISPF Dialog Manager Table handling to provide a full member selection list for a PDS data set. It then processes selected members. Depending on the call parameters, it updates a sequential file with IEBCOPY control statements, or constructs the dialog variable &MEMSEL to reflect the selections (in the format expected by the MEMBERS operand of the FTSSEND subcommand of NMCMD). Members may be selected for file transfer with an S to copy the member, or an R to copy with rename, in which case a new member name must be supplied. The following parameters may be passed to NMFTMSEL:

DD1=*ddname1*

Provides the DD name of the source data set for the file transmission. This indicates the data set from which the member selection list is built. If not present, a DD name of SYSUT1 is assumed.

DD2=*ddname2*

Provides the DD name of the target data set for an IEBCOPY unload operation (usually the staging data set). If not specified, a DD name of SYSUT2 is assumed.

CTL=*ddname3*

Provides the DD name of a data set that can be opened for update to contain the IEBCOPY control statements generated following user selections.

If not supplied, no data set is assumed. The dialog variable &MEMSEL contains the selections in the format expected by the MEMBERS operand of the FTSSEND subcommand of NMCMD.

MEM=*member*

May contain a single member name (for a single member unload operation) or an asterisk (*), which indicates that all members are selected. In either case, if a blank *member* value is entered, then a member selection list is presented.

Dialog Panels

NMFTFR

The FROM panel, which prompts for source data set and system details, and is displayed on entering the system by procedure NMFTS.

NMFTTO

The TO panel, which requests target data set and system details, and is displayed by procedure NMFTS.

NMFTNEW

This is displayed by procedure NMFTS when the target data set is specified to have a disposition of NEW.

NMFTMSEL

This is used by the program NMFTMSEL to display the member selection list.

NMFTSMM

This is used by procedure NMFTSM to allow the entry of multiple member names when option M is selected for this request.

NMFTVSE

This is a confirmation panel. When a VSAM source file is requested to be transmitted to a new target file, an IDCAMS *temporary export* operation is assumed. This panel allows confirmation of this request before proceeding.

NMFTVSR

This is displayed by NMFTSC when a source VSAM data set is being transmitted to an existing target VSAM data set to allow the entry of key ranges restricting the records selected.

NMFTDSTG

This is displayed by procedure NMFTSC when allocation of a new staging data set fails because it already exists. Options are provided to continue with the dialog and retry the operation, delete the data set and retry, or leave the data set and try allocating a different one.

NMFTWAIT

This is displayed at your terminal when some delay can be expected while the transmission is scheduled.

NMFTSERR

This is displayed when the transmission request is aborted due to error and displays any messages returned.

NMFTSOK

This is displayed when the transmission was scheduled successfully.

NMFTH*xxx*

These are the help panels.

Dialog Messages

NMFTM00, NMFTM01, and NMFTM02 contain ISPF messages used by the dialog.

NCL Procedures

The following NCL procedures are used by the ISPF Dialog. Each is invoked from the FRCMD or TOCMD operand on the NMCMD command. This means they are executed on successful completion of the requested transmission in the From, or sending, end and the To, or receiving, end, respectively.

$@COPY

This is invoked from $@L or $@M (see below) after successfully completing the staged transmission of an unloaded PDS data set. This procedure is executed to reload the transmitted members into the actual target data set and calls UTIL0009 to attach IEBCOPY.

$@DELETE

This is invoked through FRCMD to delete a staging data set in the transmitting system and notify the TSO user of the successful transmission completion.

$@E

This is invoked through TOCMD to import a VSAM data set from a staging file that was transmitted as a temporary exported data set.

$@L

This is invoked through TOCMD to reload a PDS from a staging data set. This procedure is used when member replacement was not requested and sets a parameter before calling procedure $@COPY to perform the operation.

$@M

This is invoked through TOCMD to reload a PDS from a staging data set. This procedure is used when member replacement was requested and sets a parameter before calling procedure $@COPY to perform the operation.

$@NMFTS

A tailoring procedure that is called from several other NCL procedures to set any installation data set naming options.

$@NOTIFY

This is invoked through FRCMD to notify the TSO user of the successful completion of the transmission.

$@R

This is invoked through TOCMD to reload a VSAM file following a staged transmission through the standard IDCAMS REPRO function. This procedure allows record replacement and merely sets a parameter before calling procedure $@REPRO to perform the operation.

$@REPRO

This is invoked from $@R or $@S and calls UTIL0007 to attach the IDCAMS system utility and perform a REPRO load operation.

$@S

This is invoked through TOCMD to reload a VSAM file following a staged transmission through the standard IDCAMS REPRO function. This procedure does not allow record replacement and sets a parameter before calling procedure $@REPRO to perform the operation.

Utilities

The dialog components are supported by the UTIL0007 and UTIL0009 utilities.

UTIL0007

Is a utility program called from NCL that attaches the system utility IDCAMS and passes control statements from NCL as parameters.

UTIL0009

Is a utility program called from NCL that attaches the system utility IEBCOPY and may override the default ddnames through parameters supplied from NCL.

Note: For more information about the utilities, see the documentation in the utility source in the CC2DSAMP data set.

Configure the Dialog

After installing the components into the correct execution libraries, you must configure the dialog. The ISPF Dialog procedure NMFTSID and the NCL procedure $@NMFTS are available for this purpose.

Procedure NMFTSID

This procedure is executed when the dialog is entered for the first time and is used to set various installation options and describe the operational environment of the region in which it is running. Parameters set by this initialization procedure are placed into the shared variable pool where they may be accessed by other components in the dialog. The following variables may be set by the installation:

SYSID

Must contain the name by which the local region (on which this procedure will run) is to be known. For example, if the dialog is to be used on two regions, commonly known as PROD and TEST, then a copy of this procedure must be available on each region with &SYSID set to PROD and TEST, as required.

LCLPREF

Contains the data set prefix to be used for any temporary files, such as staging data sets and IEBCOPY SYSIN control files, required by the application on this local system. By default, this is set to the TSO user's standard TSO prefix but may be changed to any suitable prefix and can contain more than one qualifying name (for example, SYSFTS.&SYSUSER). Because certain unrecoverable transmission failures can result in temporary data sets not being deleted, it may be useful to provide an initial prefix in this manner that clearly identifies this type of data set for later cleanup processing.

LCLUNIT

Can contain the name of a generic unit on which any local temporary data sets are placed. It is set to SYSDA by default.

LCLVOL

Can contain the name of a specific volume on which any local temporary data sets are placed.

REMPREF

Contains the data set prefix used for any temporary files, such as staging data sets, required by the application on a remote system. By default this is set to the TSO user's standard TSO prefix but can be changed to any suitable prefix and can contain more than one qualifying name (for example, SYSFTS.&SYSUSER). Because certain unrecoverable transmission failures can result in temporary data sets not being deleted, it may be useful to provide an initial prefix in this manner that clearly identifies this type of data set for later cleanup processing.

REMUNIT

Can contain the name of a generic unit on which any remote temporary data sets are placed. It is set to SYSDA by default.

&REMVOL

Can contain the name of a specific volume on which any remote temporary data sets are placed.

VSAMSPC

Can contain the cylinder allocations for staging data sets if VSAM staged file transmission are used. Specify both the primary and secondary extents as shown in the procedure where the default of (5 5) is set.

FTSTYP

Nominates the type of transmission that is used by this application and can be *system* or *private*. The default is system, which means that users must have system request privilege. Because the definitions are created by NCL procedures, users need not have system definition privilege. If the installation requires that only private requests be used, then the NCL procedure $USTSFTS must be modified.

MSGOPT

This is used to determine the type of message handling required by the dialog. For TSOE installations, we recommend that you use a default value of E, which means that the dialog traps error messages where possible and displays them on full-screen panels. A value of Y means that messages are written to the terminal followed by *** in the usual ISPF manner. A value of N suppresses message delivery.

&SYSFID*n*, &SYSTID*n*, &NMAPPL*n*, and &NMLINK*n*

Are used to inform the dialog of the valid system combinations available to the user for file transfer. Each combination, identified by *n* (which must be serially allocated from 1 onwards), requires all four variables and represents a system from which a file transmission can be requested. The variable &SYSFID*n* is the name by which a FROM or source system is known to the user (for example, PROD or TEST) and is nominated by the user when specifying the FROM system for the file transfer.

Where the SYSFID*n* used in this manner has the same value as variable &SYSID (described above), then the file transfer is deemed to be from a local system and staged transmissions are permitted. The variable &SYSTID*n* describes a *to* or target system that is accessible from &SYSFID*n*. Note that a given *from* system may have more than one target system, in which case an additional set of variables is required to describe each such combination. The variable &NMAPPL*n* provides the network name that is running on the *from* system, which can be contacted to perform the file transmission. &NMLINK*n* provides the name of the link from the system that can be used to transmit the data set to the target (that is, &SYSTID*n*) system. Code as many sets of these variables as are required for your configuration, starting with *n*=1.

Reverse (or bidirectional) links require two such definitions to allow transmission to be sourced from either end. See the procedure itself for examples of these parameters.

Procedure $@NMFTS

This NCL procedure is invoked following a staged file transmission in the source and target systems. It is used to reconstruct (if necessary) the prefix allocated to staging data sets so that the reload operation on the target data set and deletion of the staging file can proceed. It can also be used to return the console user ID when user notification is required after a remote request completes. The NCL procedure is driven with two passed parameters and must return the relevant NCL variable as described in the comments in the procedure. The second parameter passed is the TSO user ID of the user who requested the transmission, which may be useful in determining the data set prefix value to return. The following variables can be requested:

PREFIX

This is the staging data set prefix for this user.

SYSOPER

This is the name of the user ID on this system, which can be used to establish a ROF session with the system that is running on the system where the TSO user originally requested the transfer. Unless completion notification is required after requesting a transmission from a remote system, this parameter need not be set. (Notification following a transfer from the local system is always available). The console user ID returned in this parameter must be able to issue the ROUTE command, to connect to the system where the TSO user originated the request, and on that system issue an OPSYS command to send the necessary notification to the user.

Appendix A: Health Checks

This section contains the following topics:

CA Health Checker

The CA Health Checker provides a simple and consistent method for CA products to create health checks to run under the IBM Health Checker for z/OS. The IBM Health Checker for z/OS helps you identify potential problems in your z/OS environment by checking system or product parameters and system status against recommended settings. CA SOLVE:FTS health checks are automatically activated on the target system when the product is started on a system with IBM Health Checker for z/OS installed and configured.

The CHECK_OWNER for all CA SOLVE:FTS health checks is CA_NM.

Use either CA SYSVIEW or SDSF Health Checker displays to list and view the checks. View messages generated by CA health checks in the MVS System Log.

NM_ACB

Description

This CA SOLVE:FTS health check checks that the primary ACB of the region is open. This check runs every 5 minutes.

Best Practice

VTAM is required to access the 3270 interface. If you primarily use the WebCenter interface to access you region, you can lower the priority of this health check.

Parameters accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

None.

Non-exception Messages

The following messages can appear in health checker:

- This region's primary ACB, *acbname*, is open.
- The region is shutting down. Check is not relevant at this time.

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0106E This region's primary ACB, *acbname*, is not open.

NM_INITIALIZATION

Description

This CA SOLVE:FTS health check checks region initialization. The check runs once at region startup. If an exception occurs, the check repeats every 5 minutes until initialization is successful.

Best Practice

Follow the Install Utility procedures in the *Installation Guide* to set up your region, and ensure that the parameters are specified correctly.

Parameters Accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

See the online help for region parameter groups.

Non-exception Messages

The following messages can appear in health checker:

- The region has initialized successfully.
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0104E Initialization errors have occurred in region *regionname*.

NM_SOCKETS

Description

This CA SOLVE:FTS health check checks that the sockets are available to support IP connections. The check runs every 15 minutes.

Best Practice

To help ensure IP connections, the port number for the connection must be specified and not in use by another task.

Parameters Accepted

None.

Debug Support

No.

Verbose Support

No.

Reference

None.

Non-exception Messages

The following messages can appear in health checker:

- Sockets are configured and active. HTTP port is *nnnn* URL is http://*nnn.nnn.nnn.nnn:nnnn*
- The region is initializing. Check is not relevant at this time.
- The region is shutting down. Check is not relevant at this time

Exception Messages

If an exception occurs, the following messages are issued as WTOs and written to the SYSLOG:

- CAH0001E The check timed out while waiting for a response to a command.
- NMH0110E TCP/IP interface is not active, status is *cccccccc*.
- NMH0111E No port number has been specified for this region.

Index

$

&

A

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W