Connect:Direct®
for z/OS

Facilities Guide

Version 4.8

Sterling Commerce
An IBM Company

Connect:Direct for z/OS Facilities Guide
Version 4.8
First Edition

STERLING COMMERCE SOFTWARE

*****TRADE SECRET NOTICE*****

Sterling Commerce, Inc.

4600 Lakehurst Court Dublin, OH 43016-2000 *
614/793-7000

Contents

Chapter 1

About Connect:Direct for z/OS Facilities

Connect:Direct links technologies and moves information between networked systems and computers. It manages high-performance transfers by providing:

- Automation
- Reliability
- Efficient use of resources
- Application integration
- Ease of use

Connect:Direct software offers choices in communications protocols, hardware platforms, and operating systems. It provides the flexibility to move information among mainframes, midrange systems, desktop systems, and LAN-based workstations.

In addition to the core Connect:Direct product, the following facilities are provided to fill specific needs.

- The Activity Reporting System (ARS) works with Connect:Direct for z/OS to produce reports of Connect:Direct activity using the SAS System.
- The Console Operator interface enables you to issue all Connect:Direct for z/OS commands from a z/OS console by using a MODIFY command.
- Event Services Support (ESS) generates events asynchronously in Connect:Direct and can be used by external management and automated operations applications that require real-time notification of Connect:Direct activities.
- The Connect:Direct Spool Transfer facility enables you to transfer Job Entry Subsystem (JES) spool files by copying:
  - From JES Spool Files
  - To JES Print Queues
  - To JES Reader Queues

These facilities are the subject of this guide.

Connect:Direct for z/OS Documentation

See the *Connect:Direct for z/OS Release Notes* for a complete list of the product documentation.

About This Guide

The *Connect:Direct for z/OS Facilities Guide* is for programmers and network operations staff who install, configure, and maintain the Connect:Direct for z/OS product, and for operations staff who manage and monitor Process status and data transfer with Connect:Direct for mainframe platforms.

Depending on the facility being used, this manual assumes knowledge of the following components:

- IBM z/OS operating system
- Job Control Language (JCL)
- IBM Time Sharing Option/Interactive System Productivity Facility (TSO/ISPF)
- LRS VTAM Printer Support
- VPS/CDI option

For information about the z/OS operating system, refer to the IBM z/OS documentation. For information about a particular component or environment, refer to the library of manuals for the specific products.

Task Overview

Connect:Direct for z/OS facilties include the following activities:

Task	For More Information, See
Planning and Using the Activity Reporting System	Chapter 2, *Activity Reporting System* Chapter 3, *ARS Report Examples* Chapter 4, *Requesting ARS Reports Using Screens* Chapter 5, *Requesting Multiple ARS Reports or Scheduled Processing* Chapter 6, *ARS Record Layouts*
Issuing Connect:Direct commands from a z/OS console by using a MODIFY command	Chapter 7, *Connect:Direct for z/OS Operator Interface*
Generating events asynchronously in Connect:Direct for real-time notification of Connect:Direct activities on behalf of external management and automated operations applications	Chapter 8, *Event Services Support* Chapter 9, *Using Event Services Support* Chapter 10, *Issuing Event Services Commands* Chapter 11, *Using ESS with the CICS API* Chapter 12, *Event Services Record Descriptions*
Transferring Job Entry Subsystem (JES) spool files	Chapter 13, *Spool Transfer Facility* Chapter 14, *Receiving Spool Output* Chapter 15, *Transferring Data to the JES Reader or Spool*

Chapter 2

Activity Reporting System

The Activity Reporting System (ARS) produces reports of Connect:Direct activity. While Connect:Direct itself produces statistics, ARS enables you to access more information and also provides sorting capabilities.

You can request ARS reports in three ways:

- Make requests through ARS screens using IBM Time Sharing Option/Interactive System Productivity Facility (TSO/ISPF)
- Request reports automatically through a Connect:Direct Process
- Schedule a batch job through a job scheduling subsystem

With ARS, you do not have to pass data to z/OS system management facilities (SMF) or write SAS requests to print Connect:Direct activity reports. Use ARS as a standard reporting format throughout a network for tracking Connect:Direct activity.

Note: ARS requires a SAS base running under z/OS. Connect:Direct for z/OS must be active in order to run ARS.

ARS and Connect:Direct

Connect:Direct automatically collects statistical data about Connect:Direct activity and stores it in a data set. ARS can access this statistical data and produce the following reports for a selected time period:

- Connect:Direct Activity Report lists activity by Process step
- Connect:Direct Summary Report summarizes all activity
- Connect:Direct Exception Report lists all Process steps that do not complete successfully
- Connect:Direct Security Violations Report lists security violations by signon security failures, Process security failures, and data set access security failures
- Connect:Direct Function Reports provides the activity reports for the following types of Process steps:
  - Non-Partitioned Data Set (PDS) COPY

- PDS COPY
- RUN JOB
- RUN TASK
- Submit Within a Process

Requesting ARS Reports

Following are two methods for requesting ARS reports:

- ✦ Through the ARS screens where you can create a job stream that requests reports. The screens automatically build the job or allow you to edit the sample report job stream.
- ✦ By using the sample JCL member that is provided with ARS.

Business Solutions Using ARS

ARS is designed to provide data center management with additional tools to monitor and track Connect:Direct usage. Use the information in the reports to track attempted security violations, analyze capacity planning related data, examine Connect:Direct utilization, and isolate failed Connect:Direct Processes.

The following table describes how you can use ARS.

Function	Description
Data Center Management	The data center manager wants to keep track of Connect:Direct activities performed. The Connect:Direct Activity Report lists all of the Process steps that occurred in this data center during a specified time period. The Connect:Direct NonPDS Copy Report and Connect:Direct PDS Copy Report provide additional information on all COPY steps performed.
Security Violations	A system administrator requests a listing of any Connect:Direct-related security violations on a daily basis. The Connect:Direct Security Violations Report lists the following violations: ◆ Sign on security failures ◆ Processes not run due to lack of authorization ◆ Data set access security violations
Capacity Planning	The capacity planner at a data center needs to know how many bytes are transferring between this data center and other data centers and the total transmission times. The Connect:Direct Summary Report provides this type of information by remote data center for a specified time period.

Function	Description
Connect:Direct Utilization	The data center administrator studies Connect:Direct utilization by examining how many application programs are submitted or invoked using Connect:Direct. The administrator uses this information with a job scheduling system to produce a comprehensive analysis of all jobs run at a data center. Two ARS reports provide this type of utilization summary. The Connect:Direct Run Job Report provides information about all jobs submitted by Connect:Direct. The Connect:Direct Run Task Report provides information about all programs executed under the control of Connect:Direct.
Problem Isolation	The Connect:Direct Exception Report is an excellent tool for researching why a Process does not run. This report lists each failed Process for a requested time period along with the reason for the failure.

Chapter 3

ARS Report Examples

This chapter explains the types of ARS reports available. The reports are as follows:

- Connect:Direct Activity Report
- Connect:Direct Summary Report
- Connect:Direct Exception Report
- Connect:Direct Security Violations Report
- Connect:Direct Function Reports:
  - Connect:Direct NonPDS Copy Report
  - Connect:Direct PDS Copy Report
  - Connect:Direct Run Job Report
  - Connect:Direct Run Task Report
  - Connect:Direct Submit Within a Process Report

You can request these reports in the following ways:

- Through ARS screens using TSO/ISPF
- Through a Connect:Direct Process that is defined to automatically request reports
- Through a pre-defined batch job stream that is run by a job scheduling subsystem

Connect:Direct Activity Report

The Connect:Direct Activity Report lists Connect:Direct activity by Process step. The ARS software sorts the steps for each secondary node (SNODE) location by user ID and then by Process number, for a specified time period.

You can specify the start date and time and the stop date and time. Following is an example of the Connect:Direct Activity Report.

```
Connect:Direct ACTIVITY REPORT    (ARS00001)    17:12 WEDNESDAY, JUNE 24, 2005    1
 NODE = CD.BOSTON
-------------------------------------------------------SNODE=CD.BOSTON-----------------------------------------------------
USERID      PROC      PROC      STEP          FUNCTION   ELAPSE        PNODE         CC       EXC      MSG         DATE
              #       NAME      NAME                     TIME                                          ID

ALPHA         46      COPY1     STEP01        COPY       0:00:01.89    CD.BOSTON    00000000          SCPAO00I  06/17/2005
ALLEN         63      COPY2     STEP01        COPY       0:00:02.08    CD.CHICAGO   00000000          SRPA000I  06/17/2005
BETA          87      RUNJOB    LAB           RUNJOB     0:00:01.26    CD.CHICAGO   00000000          SRJAOOOI  06/17/2005
ETHAN         89      RUNJOB    SUBSTEP       RUNJOB     0:00:00.68    CD.JERSEY    00000000          SSJA000I  06/17/2005
LEWIS          9      RUNT      STEP1         RUNTASK    0:00:00.11    CD.JERSEY    00000000          SRTA000I  06/17/2005
LOUIS         95      SUB       SUBSTEP       SUBMIT     0:00:01.34    CD.JERSEY    00000008          SSUBOOI   06/17/2005
MULLEN        96      COPY05    STEP1         COPY       0:00:01.57    CD.MIAMI     80013000          SCPA000I  06/17/2005
WARD          99      COPY05    STEP1         COPY       0:00:02.06    CD.MIAMI     80806000          SCPAOOOI  06/17/2005
WARREN       379      SUB       SUBSTEP       SUBMIT     0:00:00.96    CD.MIAMI     00000008  ***     SSUB005I  06/17/2005
-------------------------------------------------------SNODE=CD.JERSEY-----------------------------------------------------
JONES        544      RUNT      STEP1         RUNTASK    0:00:00.18    CD.BOSTON    80806000  ***     SRTA004I  06/18/2005
-------------------------------------------------------SNODE=CD.MIAMI------------------------------------------------------
USERID      PROC      PROC      STEP          FUNCTION   ELAPSE        PNODE         CC       EXC      MSG         DATE
              #       NAME      NAME                     TIME                                          ID

ALLEN        413      COPY05    COPY01        COPY       0:00:06.96    CD.BOSTON    00000000          SCPAO00I  06/17/2005
ALLEN        414      RUNTASK   LABRNJ2       RUNJOB     0:00:01.13    CD.BOSTON    00000000          SRJA000I  06/17/2005
BURNS        415      RUNT      STEP1         RUNTASK    0:00:00.28    CD.BOSTON    00000000          SRTAOOOI  06/17/2005
BURNS        418      SUB       SUBSTEP       SUBMIT     0:00:01.79    CD.BOSTON    00000000          SSUB000I  06/17/2005
GEORGE       420      PDSCOPY   COPY01        COPY       0:00:05.42    CD.BOSTON    00000000          SCPA000I  06/17/2005
JONES        421      PDSCOPY   COPY01        COPY       0:00:03.21    CD.BOSTON    00000008  ***     SDE5078I  06/17/2005
LISA         422      RUNJOB    LABRNJ2       RUNJOB     0:00:00.34    CD.BOSTON    80013000  ***     SVGS010I  06/17/2005
MILLER       425      RUNT      STEP1         RUNTASK    0:00:00.18    CD.BOSTON    80806000  ***     SRTAO04I  06/17/2005
RYAN         426      SUB       SUBSTEP       SUBMIT     0:00:00.68    CD.BOSTON    00000008  ***     SSUB005I  06/17/2005
RYAN         427      RUNJOB    LABRNJ2       RUNJOB     0:00:00.39    CD.BOSTON    80013000  ***     SVSG010I  06/19/2005
RYAN         428      RUNJOB    LABRNJ2       RUNJOB     0:00:01.54    CD.BOSTON    00000000  ***     SRJA000I  06/19/2005
RYAN         429      RUN10     STEP1         RUNTASK    0:00:00.25    CD.BOSTON    80806000  ***     SRTA004I  06/19/2005
SMITH        430      RUN20     STEP1         RUNTASK    0:00:00.09    CD.BOSTON    00000000          SRTA000I  06/19/2005
SMITH        431      SUBACCT   SUBSTEP       SUBMIT     0:00:01.19    CD.BOSTON    00000004  ***     SBPA016I  06/19/2005
TURNER       432      SUBSTAT   SUBSTEP       SUBMIT     0:00:01.59    CD.BOSTON    00000008  ***     SPQG002I  06/19/2005
WARREN       433      SUBSALE   SUBSTEP       SUBMIT     0:00:02.47    CD.BOSTON    00000000  ***     SSUB000I  06/19/2005
```

The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined.
SNODE	Connect:Direct secondary node.
USERID	Connect:Direct user ID.
PROC #	Connect:Direct Process number.
PROC NAME	Connect:Direct Process name.
STEP NAME	Name (or label) assigned to the Process step.
FUNCTION	Connect:Direct activity. Valid Process steps are COPY, RUN JOB, RUN TASK, and SUBMIT (within a Process).
ELAPSE TIME	Function duration time in hh:mm:ss.hh format.
PNODE	Connect:Direct primary node.
CC	Connect:Direct completion code in hexadecimal format.
EXC	Exception field is displayed “***” when a Process step does not complete successfully.
MSG ID	Connect:Direct message ID.
DATE	Date Connect:Direct activity began. Date format is mm/dd/yyyy. Process start date does not conform to the DATEFORM initialization parameter.

Connect:Direct Summary Report

The Connect:Direct Summary Report summarizes activity for the node containing the statistics file being examined. The report categorizes Connect:Direct activity by SNODE for a specified time period. The conclusion of the report includes a summary for all SNODEs.

Information includes:

- Total Process steps run
- Total COPY steps run
- Total RUN JOB steps run, and RUN JOB steps submitted at this node
- Total RUN TASK steps run, RUN TASK steps executed at this node
- Total SUBMIT (within a Process) steps run and number of Processes submitted at this node
- Number of successful steps and unsuccessful steps
- Total bytes tested, written, sent, and received by this node
- Total elapsed time for all Process steps in this time period
- Average time to complete a Process step
- Total elapsed time for all COPY steps
- Average time to complete a COPY step
- Effective send and receive rates

You can specify the start date and time and the stop date and time.

Following is an example of the Connect:Direct Summary Report

```
CD SUMMARY REPORT       (ARS00002)
 NODE = CD.JERSEY
 SNODE = ALL SNODES
FIRST FUNCTION BEGAN EXECUTING ON    06/17/2005 AT 10:07:03.28
LAST FUNCTION FINISHED EXECUTION ON   07/10/2005 AT 11:26:42.75
66 PROCESS STEPS WERE COMPLETED FOR THIS TIME PERIOD.
51 TOTAL COPY STEPS OR 77.3      % OF TOTAL STEPS RUN WERE COPIES
 9 TOTAL RUNJOB STEPS OR 13.6    % OF TOTAL STEPS RUN WERE RUNJOBS
 6 RUNJOBS WERE SUBMITTED ON CD.JERSEY
 3 TOTAL RUNTASK STEPS OR 4.5    % OF TOTAL STEPS RUN WERE RUNTASKS
 2 RUNTASKS WERE ATTACHED ON CD.JERSEY
 3 TOTAL SUBMIT STEPS OR 4.5     % OF TOTAL STEPS RUN WERE SUBMITS
 2 SUBMITS (WITHIN A PROCESS) ON CD.JERSEY
48 SUCCESSFUL STEPS OR 72.7      % OF TOTAL STEPS RUN COMPLETED WITH A ZERO RETURN CODE
18 UNSUCCESSFUL STEPS OR 27.3    % OF TOTAL STEPS RUN COMPLETED WITH A NON-ZERO RETURN CODE

13:22 FRIDAY, JUNE 19, 2005       2
CD SUMMARY REPORT     (ARS00002)
NODE = CD.JERSEY
SNODE = ALL SNODES
28,001,880           TOTAL BYTES WERE READ
28,019,820           TOTAL BYTES WERE SENT
28,250,480           TOTAL BYTES WERE WRITTEN
28,269,124           TOTAL BYTES WERE RECEIVED
0:26:23.98  WAS THE TOTAL "FUNCTION" TIME (SPENT PERFORMING FUNCTIONS)
0:00:24.00  WAS THE AVERAGE TIME TO COMPLETE A FUNCTION
0:26:06.91  WAS THE TOTAL "COPY" TIME (SPENT TRANSFERRING DATA)
0:00:30.72  WAS THE AVERAGE TIME TO COMPLETE A COPY
17.999 BYTES/SECOND = EFFECTIVE SEND RATE
18.16 BYTES/SECOND = EFFECTIVE RECEIVE RATE
```

The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined.
SNODE	Connect:Direct secondary node.
FIRST FUNCTION	Actual date/time that first Process step began executing during specified time period. Includes any step that started before specified time, but is under way or completes during the time period.
LAST FUNCTION	Actual date/time last Process step finished executing during specified time period. If a step started during specified time period but ended after requested stop time or stop date, it is not included in this report.
PROCESS STEPS COMPLETED	Total number of Process steps completed in time period.
TOTAL COPY STEPS	Total number of COPY steps run.
% TOTAL STEPS RUN WERE COPIES	Percentage of total steps run that are COPY steps.
TOTAL RUNJOB STEPS	Total number of RUN JOB steps run.
% TOTAL STEPS RUN WERE RUNJOB	Percentage of total steps run that are RUN JOB steps.
# RUNJOBS SUBMITTED	Number of RUN JOB steps submitted to run at “NODE.”
TOTAL RUNTASK STEPS	Total RUN TASK steps run.
% TOTAL STEPS RUN WERE RUNTASKS	Percentage of total steps run that are RUN TASK steps.
# RUNTASKS ATTACHED	Number of RUN TASK steps that are attached on “NODE.”
TOTAL SUBMIT STEPS	Total number of SUBMIT steps run.
% TOTAL STEPS RUN WERE SUBMITS	Percentage of total steps run that are SUBMIT (within a Process) steps.
# SUBMITS (WITHIN A PROCESS)	Number of SUBMIT (within a Process) steps submitted to run at “NODE.”
SUCCESSFUL STEPS	Total number of successful steps.
% OF TOTAL STEPS RUN WITH ZERO RETURN CODE	Percentage of total successful steps.
UNSUCCESSFUL STEPS	Total number of unsuccessful steps.
% OF TOTAL STEPS RUN WITH NON-ZERO RET CD	Percentage of total unsuccessful steps.
TOTAL BYTES WERE READ	Total number of bytes read by “NODE.”
TOTAL BYTES WERE SENT	Total number of bytes sent by “NODE.”

Report Field	Description
TOTAL BYTES WERE WRITTEN	Total number of bytes written by “NODE.”
TOTAL BYTES WERE RECEIVED	Total number of bytes received by “NODE.”
WAS TOTAL “FUNCTION” TIME	Sum of elapsed times of all individual Process steps. The elapsed time for this calculation is the time period between when the step started and when the step completed.
WAS AVERAGE TIME TO COMPLETE A FUNCTION	Total function time divided by number of Process steps run.
WAS TOTAL“COPY” TIME	Sum of elapsed times for all individual COPY steps. The elapsed time for this calculation is the time period between when the step started and when the step completed.
WAS AVERAGE TIME TO COMPLETE A COPY	Total COPY time divided by the total number of COPY steps run.
EFFECTIVE SEND RATE	Total bytes sent divided by elapsed time. For this calculation, the elapsed time is the time between when the first COPY step in the data set began and the last COPY step in the data set completed.
EFFECTIVE RECEIVE RATE	Total bytes received divided by elapsed time. For this calculation, the elapsed time is the time between when the first COPY step in the data set began and the last COPY step in the data set completed.

Connect:Direct Exception Report

The Connect:Direct Exception Report lists Process steps that did not complete successfully for a specified time period. The ARS software sorts Process steps by SNODE, by user ID, and then by Process number. Each item in this report displays information unique to the executed Process step. For example, the type of information that is displayed for a failed COPY step differs from that of a failed RUN TASK step. A summary of all exception cases is listed at the conclusion of the report.

```
                                                                                17:12 WEDNESDAY, JUNE 24, 2005    1
                                       CD EXCEPTION REPORT       (ARS00003)
                                          NODE = CD.BOSTON
USERID   PROC *        PROC NAME  STEP NAME     EXCEPTION INFORMATION
******   *******       *********  *********     ************************************************************************
BYRAN        424       NONCOPY    COPY01        ** NON-PDS COPY **
                                                 06/17/2005   PNODE = CD.BOSTON      SNODE = CD.JERSEY
                                                 COMPLETION CODE = 00000008          MSG ID = SDE5708I
                                                 SENDING NODE ==> CD.JERSEY
                                                 SRC DSNAME ==> USERHLQ.NONPDS
                                                 RECEIVING NODE ==> CD.BOSTON
                                                 DEST DSNAME ==> USERHLQ.NEWNONPDS
                                                 CATALOG ERROR: CATALOG STRUCTURE OR NEW REQUEST FAILED
-------------------------------------------------------------------------------------------------------------------
JONES        421       PDSCOPY    COPY01        ** PDS COPY **
                                                 06/17/2005   PNODE = CD.BOSTON      SNODE = CD.JERSEY
                                                 COMPLETION CODE = 00000008          MSG ID = SDE5708I
                                                 SENDING NODE ==> CD.JERSEY
                                                 SRC DSNAME ==> USERHLQ.TESTPDS
                                                 RECEIVING NODE ==> CD.BOSTON
                                                 DEST DSNAME ==> USERHLQ.NEWPDS
                                                 CATALOG ERROR: CATALOG STRUCTURE OR NEW REQUEST FAILED
-------------------------------------------------------------------------------------------------------------------
MILLER       425       RUNJOB     LABRNJ2       ** RUNJOB **
                                                 06/17/2005   PNODE = CD.BOSTON      SNODE = CD.JERSEY
                                                 COMPLETION CODE = 80013000          MSG ID = SVSG010I
                                                 DATASET CONTAINING JOB = USERHLQ.CNTLIMEMBER
                                                 INTENDED NODE FOR JOB SUBMISSION = CD.BOSTON
                                                 NON-VSAM OPEN ERROR OCCURRED
-------------------------------------------------------------------------------------------------------------------
SMITH        479       RUNT       STEP1         ** RUNTASK **
                                                 06/17/2005   PNODE = CD.BOSTON      SNODE = CD.JERSEY
                                                 COMPLETION CODE = 80806000          MSG ID = SRTA004I
                                                 PROGRAM NAME = ABCNAME
                                                 INTENDED NODE FOR PROGRAM EXECUTION = CD.JERSEY
                                                 Connect:Direct RUN TASK CANNOT FIND SPECIFIED TASK IN STEP LIBRARY
-------------------------------------------------------------------------------------------------------------------
WARD         499       SUB        SUBSTEP       ** SUBMIT **
                                                 06/17/2005   PNODE = CD.BOSTON      SNODE = CD.JERSEY
                                                 COMPLETION CODE = 00000008          MSG ID = SSUB005I
                                                 DATASET CONTAINING PROCESS = THERE IS NO SUCH
                                                             LIBRARY AS THIS.(NOEXIST)
                                                 INTENDED NODE FOR PROCESS SUBMISSION = CD.JERSEY
                                                 UNABLE TO ALLOCATE DSN SPECIFIED IN SUBMIT STATEMENT


                                                                                17:12 WEDNESDAY, JUNE 24, 2005    3
                                       CD EXCEPTION REPORT       (ARS00003)
                                          NODE = CD.BOSTON
SUMMARY INFORMATION
**************************
9 NON-SUCCESSFUL STEPS IN 17 PROCESS STEPS RUN OR
--> 52.9 % OF TOTAL PROCESS STEPS RUN COMPLETED WITH A NON-ZERO RETURN CODE
1 NON-SUCCESSFUL COPIES OF 3 COPY STEPS RUN OR
--> 33.3 % OF TOTAL COPY STEPS RUN COMPLETED WITH A NON-ZERO RETURN CODE
2 NON-SUCCESSFUL RUNJOBS IN 4 RUNJOB STEPS RUN OR
--> 60 % OF TOTAL RUNTASK STEPS RUN COMPLETED WITH A NON-ZERO RETURN CODE
3 NON-SUCCESSFUL SUBMITS IN 5 SUBMIT STEPS RUN OR
--> 60 % OF TOTAL SUBMIT STEPS RUN COMPLETED WITH A NON-ZERO RETURN CODE
```

You can specify the start date and time and the stop date and time. The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined.
USERID	Connect:Direct user ID.
PROC #	Connect:Direct Process number.
PROC NAME	Connect:Direct Process name.
STEP NAME	Name (or label) assigned to the Process step.
function type	Connect:Direct activity. Valid Process steps are PDS COPY, NONPDS COPY, RUN JOB, RUN TASK, and SUBMIT (within a Process).
mm/dd/yyyy	Data Process step began executing.

Report Field	Description
PNODE	Connect:Direct primary node.
SNODE	Connect:Direct secondary node.
COMPLETION CODE	Connect:Direct completion code in hexadecimal format.
MSG ID	Connect:Direct message ID.
SENDING NODE	(COPY) Sending Connect:Direct node location name.
SRC DSNAME	(COPY) Source data set name.
RECEIVING NODE	(COPY) Receiving Connect:Direct node location name.
DEST DSNAME	(COPY) Destination data set name.
DATASET CONTAINING JOB	(RUNJOB) Data set containing job.
INTENDED NODE FOR JOB SUBMISSION	(RUNJOB) Intended node location where the job is submitted.
PROGRAM NAME	(RUNTASK) Program name.
INTENDED NODE FOR PROGRAM EXECUTION	(RUNTASK) Intended attach node where the program is run.
DATASET CONTAINING PROCESS	(SUBMIT) Data set containing Process.
INTENDED NODE FOR PROCESS SUBMISSION	(SUBMIT) Intended node location where the Process is submitted.
message	Error message for Connect:Direct message ID.
NON-SUCCESSFUL STEPS	Total number of unsuccessful Process steps.
PROCESS STEPS RUN	Total number of Process steps run.
% TOTAL PROCESS STEPS WITH NON-ZERO CODE	Percentage of Process steps unsuccessful.
NON-SUCCESSFUL COPIES	Number of unsuccessful COPY steps.
COPY STEPS RUN	Total number of COPY steps run.
% TOTAL RUN JOB STEPS WITH NON-ZERO CODE	Percentage of RUN JOB steps unsuccessful.
NON-SUCCESSFUL RUNTASKS	Number of unsuccessful RUN TASK steps.
RUNTASK STEPS RUN	Total number of RUN TASK steps run.
% TOTAL RUNTASK STEPS WITH NON-ZERO CODE	Percentage of RUN TASK steps unsuccessful.
NON-SUCCESSFUL SUBMITS	Number of unsuccessful SUBMIT (within a Process) steps.

Report Field	Description
SUBMIT STEPS RUN	Total number of SUBMIT (within a Process) steps run.
% TOTAL SUBMIT STEPS WITH NON-ZERO CODE	Percentage of SUBMIT (within a Process) steps unsuccessful.

Connect:Direct Security Violations Report

The Connect:Direct Security Violations Report lists the following types of violations for a specified time period:

- Signon security failures—This failure is caused by an invalid user ID or password that signed to Connect:Direct. Each violation is recorded.
- Process security failures—This failure is caused when a Process does not run due to an invalid security authorization. An example is a Process that does not run because the user is not defined in the Connect:Direct authorization file at the remote node.
- Data set access security failures—This failure is due to insufficient authority to access a data set.

A sample report follows:

```
CD SECURITY VIOLATIONS REPORT            (ARS00004)                  17:12 WEDNESDAY, JUNE 24, 2005  2
 NODE = CD.BOSTON
USERID        CC            MSG         DATE       TIME              VIOLATION
                            ID                                       TYPE
JONES   00000008            RACF095I  06/17/2005  10:34.30.5        DATASET
SMITH   00000008            RACF095I  06/17/2005  10:49.21.1        DATASET
MILLER  00000008            SAFA005I  06/17/2005  12:26.10.4        SIGNON
BYRAN   00000008            RACF098I  06/17/2005  17:55.15.8        SIGNON
BYRAN   00000008            RACF098I  06/17/2005  17:56.37.5        SIGNON
BYRAN   00000008            RACF098I  06/17/2005  17:57.03.1        SIGNON
GEORGE  00000008            SAFB004I  06/17/2005  19:00.30.2        PROCESS
WARREN  00000008            SAFB004I  06/17/2005  20:00.30.2        PROCESS
```

You can specify the start date and time and the stop date and time.

Included in these reports are security message IDs generated by any security subsystem used with Connect:Direct. Security subsystems supported include:

- IBM Resource Access Control Facility (RACF)
- CA-ACF2 and CA-TOP SECRET by Computer Associates, Inc.

The following table contains a description of the report fields:

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined.
USERID	Connect:Direct user ID that created security violation.
CC	Connect:Direct completion code in hexadecimal format.

Report Field	Description
MSG ID	Security system message ID: Connect:Direct, ACF2, RACF, or TOP SECRET.
DATE	Date security violation occurred.
TIME	Time security violation occurred.
VIOLATION TYPE	Type of security violation. Valid types are SIGNON, PROCESS, and DATASET.

Connect:Direct Function Reports

The Connect:Direct Function Reports provide detailed information about specific Process steps for a specified time period. The following reports are included:

- Connect:Direct NonPDS Copy Report
- Connect:Direct PDS Copy Report
- Connect:Direct Run Job Report
- Connect:Direct Run Task Report
- Connect:Direct Submit Within a Process Report

You can request the start date and time and the stop date and time when requesting one of the Connect:Direct Function Reports.

Connect:Direct NonPDS Copy Report

The Connect:Direct NonPDS Copy Report provides information about COPY steps involving these transfers:

- NonPDS data set <--------> NonPDS data set
- NonPDS data set <--------> PDS data set member
- PDS data set member <--------> NonPDS data set

Following is an example of the Connect:Direct NonPDS Copy Report.

```
                                                                                                    17:12 WEDNESDAY, JUNE 24, 2005     1
CD NON-PDS COPY REPORT     (ARS00005)
 NODE = CD.BOSTON
USERID    PROC #        PROC NAME  STEP NAME    TRANSMISSION INFORMATION
******    ******        *********  *********    ************************************************************
JONES        511        COPY05     COPY01       06/17/2005   PNODE = CD.BOSTON            SNODE = CD.JERSEY
TRANSMISSION TIME = 0:00:06.96       COMPLETION CODE = 00000000         MSG 10 = SCPA000I
COPY STEP SUCCESSFUL
 ================================================
                                       SRC DSNAME IS USERHLQ.OLDFILE1
                                       DEST DSNAME IS USERHLQ.NEWFILE1
    SENDING NODE      ==>       CD.BOSTON   RECEIVING NODE          ==>          CD.JERSEY
      BYTES READ      ==>       37200               BYTES WRITTEN  ==>          37200
     BLOCKS READ      ==>       12                  BLOCKS WRITTEN              ==>  12
       RECS READ      ==>       0                   RECS WRITTEN   ==>          0
      BYTES READ      ==>       37224               BYTES RECEIVED              ==>  37224
    COMPRESSION%      ==>       -0.1                COMPRESSION%   ==>          -0.1
         VOLSER       ==>       ABC003              VOLSER         ==>          PRD729
-------------------------------------------------------------------------------------------------------------------------------
-
MILLER       523        COPY07     COPY01       06/17/2005   PNODE = CD.BOSTON            SNODE = CD.MIAMI
TRANSMISSION TIME = 0:00:02.17       COMPLETION CODE = 00000000         MSG 10 = SCPA000I
COPY STEP SUCCESSFUL
 ================================================
                                       SRC DSNAME IS USERHLQ.SAMPFILE(MEMBER)
                                       DEST DSNAME IS USERHLQ.COMPFILE
    SENDING NODE      ==>       CD.BOSTON   RECEIVING NODE          ==>          CD.MIAMI
      BYTES READ      ==>       1120                BYTES WRITTEN  ==>          1120
     BLOCKS READ      ==>       1                   BLOCKS WRITTEN              ==>  1
       RECS READ      ==>       0                   RECS WRITTEN   ==>          0
      BYTES READ      ==>       1120                BYTES RECEIVED              ==>  1120
    COMPRESSION%      ==>       -0.2                COMPRESSION%   ==>          -0.2
         VOLSER       ==>       DCM011              VOLSER         ==>          PRD011
-------------------------------------------------------------------------------------------------------------------------------
-MILLER       523        COPY07     COPY01       06/17/2005   PNODE = CD.BOSTON            SNODE = CD.MIAMI
TRANSMISSION TIME = 0:00:00.87       COMPLETION CODE = 00000000         MSG 10 = SCPA000I
COPY STEP SUCCESSFUL
 ================================================
                                       SRC DSNAME IS USERHLQ.FILE01
                                       DEST DSNAME IS USERHLQ.FILE02
    SENDING NODE      ==>       CD.BOSTON   RECEIVING NODE          ==>          CD.MIAMI
      BYTES READ      ==>       320                 BYTES WRITTEN  ==>          320
     BLOCKS READ      ==>       1                   BLOCKS WRITTEN              ==>  1
       RECS READ      ==>       0                   RECS WRITTEN   ==>          0
      BYTES READ      ==>       320                 BYTES RECEIVED              ==>  320
    COMPRESSION%      ==>       -0.6                COMPRESSION%   ==>          -0.6
         VOLSER       ==>       PRD011              VOLSER         ==>          DCM011
-------------------------------------------------------------------------------------------------------------------------------
-
```

This report includes all NonPDS COPY step transmissions for a specified time period. The ARS software sorts COPY information for each SNODE by user ID and then in ascending order by Process number. The following table contains a description of the report fields.

Report Field	Descriptions
NODE	Connect:Direct node where the statistics file is being examined
USERID	Connect:Direct user ID
PROC #	Connect:Direct Process number
PROC NAME	Connect:Direct Process name
STEP NAME	Name (or label) assigned to a COPY step
mm/dd/yyyy	Date COPY step began executing
PNODE	Connect:Direct primary node
SNODE	Connect:Direct secondary node
TRANSMISSION TIME	Elapsed time between when the COPY step started and when the COPY step completed
COMPLETION CODE	Connect:Direct completion code in hexadecimal format

Report Field	Descriptions
MSG ID	Connect:Direct message ID
message	Short message for Connect:Direct message ID
SRC DSNAME	Source data set name
DEST DSNAME	Destination data set name
REPORT FIELD	Description of report
SENDING NODE	Name of node sending the data set
RECEIVING NODE	Name of node receiving the data set
BYTES READ	Number of bytes read by sending node
BYTES WRITTEN	Number of bytes written by receiving node
BLOCKS READ	Number of blocks read by sending node
BLOCKS WRITTEN	Number of blocks written by receiving node (either blocks or records is displayed)
RECS READ	Number of records read by sending node
RECS WRITTEN	Number of records written by receiving node
BYTES SENT	Number of bytes sent by sending node
BYTES RECEIVED	Number of bytes received by receiving node
COMPRESSION%	Compression percentage for sending data set
COMPRESSION%	Compression percentage for receiving data set
VOLSER	Sending volume serial number
VOLSER	Receiving volume serial number

Connect:Direct PDS Copy Report

The Connect:Direct PDS Copy Report provides information for each COPY step involving the following type of transfer:

- PDS data set <--------> PDS data set

Following is an example of the Connect:Direct PDS Copy Report.

```
                                                    17:12 WEDNESDAY, JUNE 24, 2005    1
                                      CD PDS COPY REPORT        (ARS00006)
                                            NODE = CD.BOSTON
USERID      PROC #     PROC NAME   STEP NAME           TRANSMISSION INFORMATION
******      ******     *********   *********           ***************************
JONES       420        PDSCOPY     COPY01  06/17/2005  PNODE = CD.BOSTON SNODE = CD.JERSEY
TRANSMISSION TIME = 0:00:05.42               COMPLETION CODE = 00000000  MSGID = SCPAOOOI
COPY STEP SUCCESSFUL
================================================
                                     SRC DSNAME IS USERHLQ.TESTPDS
                                     DEST DSNAME IS USERHLQ.NEWPDS
   SENDING NODE        ==>       CD.BOSTON  RECEIVING NODE        ==>     CD.JERSEY
     BYTES READ        ==>       22240      BYTES WRITTEN         ==>     22240
    BLOCKS READ        ==>       11         BLOCKS WRITTEN                 ==>     11
       RECS READ       ==>       0          RECS WRITTEN          ==>     0
     BYTES SENT        ==>       22262      BYTES RECEIVED                 ==>     22262
   COMPRESSION%        ==>       -0.1       COMPRESSION%          ==>      -0.1
         VOLSER        ==>       ABC003     VOLSER                ==>     ABC013
                                 ABC005
                                 ABC007
                                 ABC009
                                 ABC011
                                     ============================================
                                          MEMBER LIST
                                     ============================================
                                          MEMBER 1   TO          MEMBER 1
                                          MEMBER 2   TO          MEMBER 2
                                          MEMBER 3   TO          MEMBER 3
                                          MEMBER 4   TO          MEMBER 4
                                          MEMBER 5   TO          MEMBER 5
                                          MEMBER 6   TO          MEMBER 6
                                          MEMBER 7   TO          MEMBER 7
                                          MEMBER 8   TO          MEMBER 8
                                          MEMBER 9   TO          MEMBER 9
                                          MEMBER 10  TO          MEMBER 10
                                          MEMBER 11  TO          MEMBER 11
                                          MEMBER 12  TO          MEMBER 12
```

This report lists all PDS COPY step transmissions by Process number for a specified time period. It lists sending data set members names with each destination data set member name.

We cannot guarantee the accuracy of the data in this report if you restart Connect:Direct using the TCQ=COLD initialization parameter during the time period specified. ARS uses the Process number in the statistics file to match PDS member names with a specific COPY step. Because a COLD restart begins numbering Processes from **1**, ARS may not associate PDS member names correctly with the appropriate PDS Copy step.

The following table shows a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined
USERID	Connect:Direct user ID
PROC #	Connect:Direct Process number
PROC NAME	Connect:Direct Process name
STEP NAME	Name (or label) assigned to a COPY step
mm/dd/yyyy	Date COPY step began executing
PNODE	Name of primary node
SNODE	Name of secondary node
TRANSMISSION TIME	Elapsed time between when the COPY step started and when the COPY step completed
COMPLETION CODE	Connect:Direct completion code in hexadecimal format

Report Field	Description
MSG ID	Connect:Direct message ID
message	Short message for Connect:Direct message ID
SRC DSNAME	Source data set name
DEST DSNAME	Destination data set name
SENDING NODE	Name of node sending the data set
REPORT FIELD	Description of report
RECEIVING NODE	Name of node receiving the data set
BYTES READ	Number of bytes read by sending node
BYTES WRITTEN	Number of bytes written by receiving node
BLOCKS READ	Number of blocks read by sending node
BLOCKS WRITTEN	Number of blocks written by receiving node (either blocks or records is displayed)
RECS READ	Number of records read by sending node
RECS WRITTEN	Number of records written by receiving node
BYTES SENT	Number of bytes sent by sending node
BYTES RECEIVED	Number of bytes received by receiving node
COMPRESSION%	Compression percentage for sending data set
COMPRESSION%	Compression percentage for receiving data set
VOLSER	Sending volume serial number
VOLSER	Receiving volume serial number
MEMBER LIST	List of sending member names and receiving member names.

Connect:Direct Run Job Report

The Connect:Direct Run Job Report lists the jobs that are submitted to the system for execution using the Connect:Direct RUN JOB statement. Jobs are listed for a specified time period. Following is an example of the Connect:Direct Run Job Report.

```
                              Connect:Direct RUNJOB REPORT (ARS00007)
                                        NODE = DLCDA
USERID  PROC # PROC NAME  STEP NAME      RUNJOB INFORMATION
******  ****** *********  *********      ********************************************************************************
DLEWI1  128    RUNJOB1    RUNJOB1        09/29/2008 PNODE = DLCDA             SNODE = DLCDA
                                         COMPLETION CODE = 00000000           MSG ID = SRJA000I
                                         C:D RUN JOB FUNCTION COMPLETED SUCCESSFULLY.
                                         ================================================================================
                                         DSNAME ==> DLEWI1.CDA.JCL(IEFBR14)
                                         JOB # ==> 32749
                                         JOB SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
DLEWI1  129    RUNJOB2    RUNJOB2        09/29/2008 PNODE = DLCDA             SNODE = DLCDA
                                         COMPLETION CODE = 00000000           MSG ID = SRJA000I
                                         C:D RUN JOB FUNCTION COMPLETED SUCCESSFULLY.
                                         ================================================================================
                                         DSNAME ==> DLEWI1.CDA.JCL(IEFBR14)
                                         JOB # ==> 32750
                                         JOB SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
DLEWI1  130    RUNJOB3    RUNJOB3        09/29/2008 PNODE = DLCDA             SNODE = DLCDA
                                         COMPLETION CODE = 00000000           MSG ID = SRJA000I
                                         C:D RUN JOB FUNCTION COMPLETED SUCCESSFULLY.
                                         ================================================================================
                                         DSNAME ==> DLEWI1.CDA.JCL(IEFBR14)
                                         JOB # ==> 32751
                                         JOB SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
```

The ARS software sorts all RUN JOB steps for each SNODE by userid and then in ascending order by Process number.

The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined
USERID	Connect:Direct user ID
PROC #	Connect:Direct Process number
PROC NAME	Connect:Direct Process name
STEP NAME	Name (or label) assigned to a RUN JOB step
mm/dd/yyyy	Date RUN JOB step submitted to run
PNODE	Connect:Direct primary node
REPORT FIELD	Description of report
SNODE	Connect:Direct secondary node
COMPLETION CODE	Connect:Direct completion code in hexadecimal format
MSG ID	Connect:Direct message ID
message	Short message for Connect:Direct message ID
DSNAME	Data set name containing job stream to be submitted
JOB #	Number assigned to job by operating system
JOB SUBMITTED ON	Connect:Direct node where the job is submitted to run

Connect:Direct Run Task Report

The Connect:Direct Run Task Report tracks all tasks (programs) for a specified time period that executed under the control of the Connect:Direct RUN TASK statement. Following is an example of the Connect:Direct Run Task Report.

```
                              Connect:Direct RUNTASK REPORT (ARS00008)
                                        NODE = DLCDA
USERID  PROC # PROC NAME  STEP NAME   RUNTASK INFORMATION
******  ****** *********  *********   ********************************************************************************
        1      RSS2C2CV   DTFA        09/25/2008 PNODE = DLCDA             SNODE = DLCDA
                                      COMPLETION CODE = 00000000           MSG ID = SRTA000I
                                      C:D RUN TASK FUNCTION SUCCESSFULLY COMPLETED.
                                      ================================================================================
                                      PROGRAM NAME  ==> DMRTSUB
                                      PROGRAM ATTACHED ON DLCDA
--------------------------------------------------------------------------------------------------------------------
        1      RSS2C2CV   DTFB        09/25/2008 PNODE = DLCDA             SNODE = DLCDA
                                      COMPLETION CODE = 00000000           MSG ID = SRTA000I
                                      C:D RUN TASK FUNCTION SUCCESSFULLY COMPLETED.
                                      ================================================================================
                                      PROGRAM NAME  ==> DMRTSUB
                                      PROGRAM ATTACHED ON DLCDA
--------------------------------------------------------------------------------------------------------------------
        1      RSS2C2CV   WAIT        09/25/2008 PNODE = DLCDA             SNODE = DLCDA
                                      COMPLETION CODE = 00000000           MSG ID = SRTA000I
                                      C:D RUN TASK FUNCTION SUCCESSFULLY COMPLETED.
                                      ================================================================================
                                      PROGRAM NAME  ==> DMRTWAIT
                                      PROGRAM ATTACHED ON DLCDA
--------------------------------------------------------------------------------------------------------------------
```

The ARS software sorts all RUN TASK steps for each SNODE by userid and then in ascending order by Process number.

The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined
USERID	Connect:Direct user ID
PROC #	Connect:Direct Process number
PROC NAME	Connect:Direct Process name
STEP NAME	Name (or label) assigned to a RUN TASK step
mm/dd/yyyy	Date RUN TASK step attached and executed
PNODE	Connect:Direct primary node
REPORT FIELD	Description of report
SNODE	Connect:Direct secondary node
COMPLETION CODE	Connect:Direct completion code in hexadecimal format
MSG ID	Connect:Direct message ID
message	Short message for Connect:Direct message ID
PROGRAM NAME	Name of program module attached
PROGRAM ATTACHED ON	Connect:Direct node where the task is attached and executed

Connect:Direct Submit Within a Process Report

The Connect:Direct Submit Within a Process Report lists Processes that are submitted to Connect:Direct for execution using the Connect:Direct Submit (within a Process) statement for a specified time period. Following is an example of the Connect:Direct Submit Within a Process Report.

```
                                Connect:Direct SUBMIT WITHIN A PROCESS RPT (ARS00009)
                                     NODE = DLCDA
USERID   PROC # PROC NAME  STEP NAME   SUBMIT INFORMATION
******   ****** *********  *********   ********************************************************************************
DLEWI1   8      DSK2VSE                09/25/2008 PNODE = DLCDA             SNODE = DLEWIS.VSE.TCP
                                       COMPLETION CODE = 00000000           MSG ID = SSUB000I
                                       THE SUBMIT CONTROL BLOCK HAS BEEN SUCCESSFULLY CONSTRUCTED.
                                       ================================================================================
                                       DSNAME ==> DLEWI1.CDA.PROCESS(DSK2VSE)
                                       PROCESS SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
DLEWI1   9      DSK2VSE                09/25/2008 PNODE = DLCDA             SNODE = DLEWIS.VSE.TCP
                                       COMPLETION CODE = 00000000           MSG ID = SSUB000I
                                       THE SUBMIT CONTROL BLOCK HAS BEEN SUCCESSFULLY CONSTRUCTED.
                                       ================================================================================
                                       DSNAME ==> DLEWI1.CDA.PROCESS(DSK2VSE)
                                       PROCESS SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
DLEWI1   10     DSK2VSE                09/25/2008 PNODE = DLCDA             SNODE = DLEWIS.VSE.TCP
                                       COMPLETION CODE = 00000000           MSG ID = SSUB000I
                                       THE SUBMIT CONTROL BLOCK HAS BEEN SUCCESSFULLY CONSTRUCTED.
                                       ================================================================================
                                       DSNAME ==> DLEWI1.CDA.PROCESS(DSK2VSE)
                                       PROCESS SUBMITTED ON DLCDA
--------------------------------------------------------------------------------------------------------------------------
```

The ARS software sorts all SUBMIT steps for each SNODE by userid and then in ascending order by Process number.

The following table contains a description of the report fields.

Report Field	Description
NODE	Connect:Direct node where the statistics file is being examined
USERID	Connect:Direct user ID
PROC #	Connect:Direct Process number
PROC NAME	Connect:Direct Process name
STEP NAME	Name (or label) assigned to a SUBMIT (within a Process) step
mm/dd/yyyy	Date SUBMIT (within a Process) step ran
PNODE	Connect:Direct primary node
REPORT FIELD	Description of report
SNODE	Connect:Direct secondary node
COMPLETION CODE	Connect:Direct completion code in hexadecimal format
MSG ID	Connect:Direct message ID
message	Short message for Connect:Direct message ID
DSNAME	Data set name containing the Connect:Direct Process. Only available when SUBNODE=PNODE.

Report Field	Description
PROCESS SUBMITTED ON	Connect:Direct node where the Process was submitted to run

Chapter 4

Requesting ARS Reports Using Screens

This chapter explains how to request ARS statistical reports using ARS screens. Request these reports by completing the following two steps:

- Step 1–Request the ARS report using ARS screens
- Step 2–Run the job

ARS screens build a job stream to produce any of the ARS reports. For this reason, the screens are most useful in cases where you need individual reports instead of volume reports.

After ARS screens build the job stream and submit it to run, the job performs the following:

- Signs on to Connect:Direct
- Creates a temporary data set to hold the specified statistics
- Accesses the Connect:Direct statistics file and copies statistics to a temporary data set
- Executes the requested ARS report routine using the Connect:Direct statistics as input data
- Sends the report to the selected output location
- Deletes the temporary data set

Step 1—Requesting Reports Using ARS Screens

The ARS screens prompt you to provide information that automatically builds the job discussed in Step 2 of this chapter. Certain information entered on these screens is retained between sessions to alleviate the need for entering identical information each time you request a report.

Accessing the ARS Screens Using the Connect:Direct IUI

To access the ARS screens through the IUI, select the ADMIN option from the Primary Options Menu.

```
              Connect:Direct PRIMARY OPTIONS MENU
CMD ==> ADMIN

SELECT ONE OF THE FOLLOWING:
   CF  - COPY A FILE                                  ******************
   SB  - SUBMIT A PREDEFINED PROCESS                  *                *
   DF  - DEFINE A PROCESS USING ISPF EDIT             * TODAY:1998.06.24*
   SS  - VIEW STATISTICS FOR A COMPLETED PROCESS      * TIME: 09:20     *
   MB - SUBMIT A BATCH TO Connect:Enterprise for z/OS *                *
   CP  - CHANGE CHARACTERISTICS OF A PROCESS          ******************
   DP  - DELETE A NON-EXECUTING PROCESS
   FP  - FLUSH AN EXECUTING PROCESS
   SP  - VIEW DATA ABOUT AN EXECUTING PROCESS
   PS  - SUSPEND AN EXECUTING PROCESS
   MSG - VIEW Connect:Direct MESSAGE TEXT
   SW  - SWAP AMONG CONCURRENT Connect:Direct SESSIONS
   SD  - VIEW/CHANGE YOUR Connect:Direct SIGNON INFORMATION DEFAULTS
   NM  - VIEW INFORMATION IN THE Connect:Direct NETWORK MAP
   WHO - VIEW CHARACTERISTICS OF YOUR Connect:Direct IUI ENVIRONMENT
   SPF - ENTER ISPF/PDF
   AUTH  - VIEW YOUR Connect:Direct FUNCTION AUTHORIZATION
   MS  - SIGN ON TO MULTIPLE Connect:DIRECT NODES CONCURRENTLY
   ADMIN - PERFORM Connect:Direct ADMINISTRATIVE FUNCTIONS
```

From the Administrative Options Menu, select the ARS option.

```
View, Modify, Control, Delete, Secure+
_____________________________________________________________________________
node.name              Connect:Direct ADMINISTRATIVE OPTIONS MENU
CMD ==>

 SELECT ONE OF THE FOLLOWING:
   ST    - VIEW TYPE RECORD                         ******************
   IT    - INSERT/UPDATE TYPE RECORD                *                *
   DT    - DELETE TYPE RECORD                       * TODAY:1998.06.24*
   SU    - VIEW USER AUTHORIZATION RECORD           * TIME: 09:23     *
   IU    - INSERT/UPDATE USER AUTHORIZATION RECORD *                  *
   DU    - DELETE USER AUTHORIZATION RECORD ******************
   TS    - VIEW Connect:Direct TASKS
   TF    - FLUSH A Connect:Direct TASK

   MD    - MODIFY Connect:Direct TRACE CHARACTERISTICS
   C     - ENTER A NATIVE Connect:Direct COMMAND
   SN    - TERMINATE Connect:Direct
   ARS   - ARS REPORTING FACILITY
   NM    - VIEW THE CONTENTS OF THE Connect:Direct NETWORK MAP
   UNM   - UPDATE THE Connect:Direct NETWORK MAP

   INQ   - INQUIRE ABOUT DTF INTERNAL STATUS
   STAT  - PERFORM STATISTICS FUNCTIONS
```

Accessing the ARS Screens

In addition to the IUI, you can access the ARS screens by using one of following methods:

- Request the ARS option defined on the ISPF Primary Option Menu

- Request the TSO COMMAND Option (Option 6) from the ISPF Primary Option Menu and type the following.

```
ARS
```

- Type the following at the command line on any ISPF screen.

```
==> TSO %ARS
```

The following table contains a description of these ARS signon parameters.

Parameters	Description of ARS Signon Parameters
TSO	(optional) specifies a TSO command. Use it when accessing ARS from the ISPF command line.
ARS	specifies the name of the CLIST that accesses the ARS screens.

ARS Report Options Screen

The ARS Report Options Screen in the following figure enables you to request the report, provide signon data, and specify the time period covered in the report.

If you already requested ARS reports using these screens and do not need to change any other data, you can bypass the remaining screens by typing SUB (submit) at the CMD field, then pressing Enter. This command automatically submits the job stream to run.

```
$cd.node         Connect:Direct - ARS REPORT OPTIONS
CMD ==>                                                                  hh:mm

        AC - ACTIVITY                            PS - NON-PDS COPY
        SM - SUMMARY                             PO - PDS COPY
        EX - EXCEPTION                           RJ - RUNJOB
        SC - SECURITY                            RT - RUNTASK
                                                 SB - SUBMIT
  REPORT TYPE  ==>          EDIT JCL ==>   (Y,N)

  Connect:Direct SIGNON PARAMETERS
  --------------------------------
  USER ID      ==> $uid
  PASSWORD     ==>
  NETMAP NAME  ==> $cd.netmap
  TRANSPORT    ==> NET
  COMMUNICATION ADDRESS ==> (      ,                 )

  REPORTING RANGE
  ---------------
  START DATE   ==>             START TIME  ==>
  STOP  DATE   ==>             STOP  TIME  ==>
```

The following table contains a description of the screen fields.

Field	Description
REPORT TYPE	Specifies the ARS report.
EDIT JCL	Specifies an option to edit JCL.
USER ID	Specifies the userid for signing on to Connect:Direct, if your security environment requires it. (optional)
PASSWORD	Specifies the password needed to access Connect:Direct. (optional)
NETMAP NAME	Specifies the name of the Connect:Direct Network Map.
TRANSPORT	Enables override of signon defaults for DMBATCH transport type.
COMMUNICATION ADDRESS	Specifies port number and IP address with TRANSPORT=TCP.
REPORTING RANGE	Identifies the date, day or time period that you want the requested report to cover. (optional) If all four fields are blank, the entire contents of the Connect:Direct statistics file is used.

Field	Description
START DATE or STOP DATE	Specifies the date or day that the statistics records are selected for the ARS report. You can specify the day (dd), month (mm), and year (yy for 2-digit year and yyyy for 4-digit year). Use periods or backslashes (\) to separate the components of a date value. You can specify the date (dd), month (mm), and year (yy) in one of the following formats: yymmdd or yyyymmdd; yy/mm/dd or yyyy/mm/dd; yy.mm.dd or yyyy.mm.dd; mmddyy or mmddyyyy; mm/dd/yy or mm/dd/yyyy; mm.dd.yy or mm.dd.yyyy; or the Julian date, yyddd or yyyyddd; yy/ddd or yyyy/ddd; or yy.ddd or yyyy.ddd. If you only specify the date, the time defaults to 00:00. This date must have the same format as specified in the DATEFORM initialization parameter. You can also use day in these fields to indicate day of the week for which the statistics records are searched. Valid names include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. You can also specify TODAY, which searches for the statistics records today; or TOMORROW, which searches for statistics records the next day; or YESTER, which searches for statistics records for yesterday.
START TIME or STOP TIME	Specifies the time of day in hours (hh), minutes (mm), and seconds (ss) when the statistics records are selected for the ARS report. Specify AM or PM. You can express the time of day using the 24-hour clock or the 12-hour clock. If you use the 24-hour clock, valid times are 00:00-24:00. If you use the 12-hour clock, you can express 1:00 hours as 1:00AM, and you can express 1:00 PM as 13:00 hours. If you do not use AM and PM, the 24-hour clock is assumed. You do not need to specify minutes and seconds. You can also specify NOON, which searches for the statistics records at noon, or MIDNIGHT, which searches for the statistics records at midnight. The default for the time is 00:00:00, the beginning of the day. If you do not specify START TIME and STOP TIME but you do specify the START DATE and STOP DATE, the time range default is 00:00 to 24:00.

ARS Connect:Direct Requirements Screen

The ARS Connect:Direct Requirements screen in the following figure identifies specific job stream parameters for the job card and Connect:Direct-related information.

```
$cd.node         Connect:Direct - ARS (Job and File Requirements)
CMD ==>                                                              hh:mm

UP TO 3 LINES FOR JOBCARD INFO
==> ____________________________________________________________________________
==> ____________________________________________________________________________
==> ____________________________________________________________________________

STEPLIB DATA SET NAME FOR Connect:Direct PROGRAMS (OPTIONAL):
==> ____________________________________________
==> ____________________________________________

Connect:Direct PUBLIB PROCESS LIBRARIES (1 REQUIRED):
==> ____________________________________________
==> ____________________________________________

Connect:Direct MESSAGE DATA SET NAME (REQUIRED):
==> ____________________________________________

UNIT SPECIFICATION FOR Connect:Direct TEMPORARY DATASET:
==> ________
```

The following table contains a description of the screen fields.

Field	Description
JOBCARD INFO	Specifies typical job card information. Use the COND=(0,NE) for best results. This means that Step 2 of the job does not run if an error is found in Step 1.
STEPLIB DATA SET NAME FOR Connect:Direct PROGRAMS	Specifies the name of the library containing the Connect:Direct load modules. You can type two libraries in this section (optional).
Connect:Direct PUBLIC PROCESS LIBRARIES	Specifies the name of the library containing Connect:Direct Processes.
Connect:Direct MESSAGE DATA SET NAME	Specifies the name of the message data set containing the Connect:Direct messages.

ARS SAS Requirements Screen

The ARS SAS Requirements screen in the following figure identifies specific JCL parameters for SAS-related information.

```
$cd.node         Connect:Direct - ARS (SAS Files and Output)
CMD ==>                                                              hh:mm

SAS CATALOGED PROCEDURE (REQUIRED):
==> _____________________________________________

DATASET NAME CONTAINING SAS ROUTINES (REQUIRED):
==> _____________________________________________

ONE OF THE FOLLOWING MUST BE PROVIDED :

OUTPUT DATA SET NAME :
==> _____________________________________________

OUTPUT (SYSOUT) CLASS :
==> _
```

The following table contains a description of the screen fields.

Field	Description
SAS CATALOGED PROCEDURE	Identifies the name of the SAS cataloged procedure, which is a collection of JCL statements required to execute SAS for batch processing at your installation.
DATA SET NAME CONTAINING SAS ROUTINES	Specifies where the ARS routines ($CD.MAPLIB) that interact with SAS are located.
OUTPUT DATA SET NAME	You must preallocate the sequential data set designated in this field with the following attributes: RECFM=FBA LRECL=240 BLKSIZE=3120 If this field is left blank, you must specify the OUTPUT (SYSOUT) CLASS field.
OUTPUT (SYSOUT) CLASS	Specifies the SYSOUT class that automatically sends the output to a designated printer queue (optional). If you leave this field blank, you must specify the OUTPUT DATASET NAME field. **Note:** If you are requesting the Connect:Direct Summary report and routing it to a data set, ensure that the data set is empty. If the data set is not empty, this report is placed after the data already in the data set. The Connect:Direct Summary Report is automatically allocated with DISP=MOD instead of DISP=SHR.

Displayed Job Stream Created from Screens

After completing the ARS screens, ARS builds a job stream, as in the following example, based on what you entered on the screens. If you requested EDIT, ARS displays the job after completing the screens.

You can review the job stream before submitting it. Following is an example of the JCL which is generated using member CDARS from $CD.ISPSLIB.

```
//JOBNAME  JOB  (ACCT), 'NAME', NOTIFY=TSOID,TIME=((1),
//REGION=1024K,MSGCLASS=X,CLASS=B
//*
//*
//DMBATCH      EXEC PGM=DMBATCH,PARM='YYSLN'
//STEPLIB      DD    DISP=SHR,DSN=$CD.LINKLIB
//SYSUDUMP     DD    SYSOUT=*
//DMPUBLIB     DD    DISP=SHR,DSN=$CD.PROCESS
//DMMSGFIL           DISP=SHR,DSN=$CD.MSG
//TEMPDSN            DSN=&CDAPI,DISP=(NEW,PASS),UNIT=PTEMP,
//                   DCB=(DSORG=PS,RECFM=VBA,LRECL=4100,BLKSIZE=4104),
//                   SPACE=(4104,(70,13))
//DMPRINT      DD    SYSOUT=*
//SYSIN        DD    *
SIGNON USERID=(NAME,,)  -
NETMAP=$CD.NETMAP TMPDD=TEMPDSN
SEL STAT WHERE (STARTT = (06/1/1998,12:00AM) -
                STOPT = 07/1/1998,11:59PM) ) FILE
SIGNOFF
/*
//*
///SASTEP EXEC $SASPROC,
//    OPTIONS='DQUOTE MACRO MACROGEN MERROR MISSING="-"'
//WORK         DD    UNIT=PTEMP,SPACE=(CYL,(20,10))
//NDMX0001     DD    DISP=(OLD,DELETE),DSN=&CDAPI
//DMMSGFIL     DD    DISP=SHR,DSN=$CD.MSG
//FT20F001     DD    DISP=SHR,DSN=$OUTPUT.DATASET.NAME
//SYSIN        DD    DISP=SHR,DSN=$CD.MAPLIB(ACTIVITY)
```

Step 2—Running the Job

If you type COND=(0,NE) in the JOBCARD INFO field of the requirements screen, ARS bypasses Step 2 if Step 1 does not complete successfully.

When the job runs, it performs the following:

- Signs on to Connect:Direct
- Creates a temporary data set to hold the specified statistics
- Accesses the Connect:Direct statistics file and copies statistics to the temporary data set
- Executes the requested ARS report routine using the Connect:Direct statistics as input data
- Sends the report to the selected output location
- Deletes the temporary data set

Chapter 5

Requesting Multiple ARS Reports or Scheduled Processing

This chapter explains how to request ARS reports without using the ARS screens. You can:

- Submit multiple ARS reports at one time
- Submit ARS reports for scheduled processing

You can request ARS reports without using ARS screens by editing a sample job stream to specify processing requirements and report types. Also, when the output of a multiple report request is routed to a data set, some ARS report routines can require a minor edit change.

This chapter contains the following:

- Sample job stream with instructions for modifying it, and definitions of the parameters and SAS options used in the job stream
- Instructions for editing the sample Connect:Direct for z/OS Process that enables you to schedule the report requested
- Instructions for editing ARS report routines when multiple reports are routed to an output data set

Sample Job Stream

The purpose of this section is to:

- Explain how to modify the sample job stream
- Define the parameters used in the job stream

The sample job stream is located in the data set named $CD.JCL(ARSJCL).

Modifying the Sample Job Stream

The following figure is a sample job stream. This job stream requests all of the ARS reports in one step. Make the following modifications of the sample job stream:

1. Modify the job card with the appropriate information.
2. Change $CD.LINKLIB to the appropriate Connect:Direct load library name.
3. Change $CD.PROCESS to the appropriate Connect:Direct Process library name.
4. Change all occurrences of $CD.MSG to the Connect:Direct message data set name.
5. Change $UID to your Connect:Direct user ID. Also add the password, if needed.
6. Change $CD.NETMAP to your Network Map data set name.
7. Change all occurrences of $UNITNAME to the valid unit name.
8. Change $SASPROC to the name of the SAS cataloged procedure used at your installation.
9. Route the output to one of the two options:
   - Route to the SYSOUT class.
   - Change all occurrences of $OUTPUT.DATASET.NAME to the name of the data set to which you route the output. Preallocate this data set as FBA with an LRECL of 240 and BLKSIZE of 3120. Ensure that the data set is empty before running this job. Complete the instructions in *Routing Multiple Reports to an Output Data Set* on page 46.

Note: This option is commented out in the sample job.

10. Change $CD.MAPLIB to the name of the data set containing the SAS programs.

    Following is an example of the ARSJCL member found in $CD.CNTL.

```
//JOBNAMEX  JOB  (ACTNG), 'PRGRMR', NOTIFY=TSOID,TIME=((1),COND=(0,NE),
//       REGION=1024k,MSGCLASS=,CLASS=,
//*********************************************************************/
//* THIS JCL CAN BE EDITED TO CREATE A JOB STREAM THAT RUNS           */
//* ARS INDEPENDENTLY OF THE ARS PANELS.  IT IS SET UP TO             */
//* PRODUCE MULTIPLE REPORTS.  FOLLOW THE INSTRUCTIONS OUTLINED BELOW.*/
//*                                                                   */
//*                                                                   */
//*  1) MODIFY JOB CARD WITH APPROPRIATE INFORMATION.                 */
//*  2) CHANGE $CD.LINKLIB TO BE THE NAME OF THE C:D LOAD LIBRARY.    */
//*  3) CHANGE $CD.PROCESS TO BE THE NAME OF THE C:D PROCESS LIBRARY. */
//*  4) CHANGE ALL OCCURRENCES OF $CD.MSG TO THE NAME OF THE C:D      */
//*      MESSAGE DATA SET.                                            */
//*  5) CHANGE $UID IN SIGNON CMD TO YOUR C:D USERID-ADD PSWD IF  NEC. */
//*  6) CHANGE $CD.NETMAP TO YOUR NETWORK MAP DATA SET NAME.          */
//*  7) CHANGE ALL OCCURRENCES OF $UNITNAME TO VALID UNIT.            */
//*  8) CHANGE $SASPROC TO THE NAME OF SAS CATALOGED PROCEDURE USED AT*/
//*     YOUR INSTALLATION.                                            */
//*  9) ROUTE OUTPUT TO A SYSOUT CLASS (LOOK FOR FT20F001 DD)         */
//*                                                                   */
//*  OR ...                                                           */
//*                                                                   */
//*  9) CHANGE ALL $OUTPUT.DATASET.NAME TO THE NAME OF THE DATA SET YOU*/
//*     HAVE CHOSEN TO ROUTE THE OUTPUT TO.  THIS DATA SET SHOULD BE  */
//*     ALLOCATED AS FBA WITH AN LRECL OF 240 AND BLKSIZE OF 3120.    */
//*     THIS DATA SET SHOULD BE EMPTY BEFORE RUNNING THIS JOB.        */
//*     (DO NOT FORGET TO DELETE THE ASTERISK ON THIS LINE IN THE JOB */
//*     AND PLACE IT ON THE LINE ABOVE BEFORE RUNNING.)               */
//*                                                                   */
//*     A) BECAUSE THE SAS PROGRAMS ARE SET UP TO BE RUN INDIVIDUALLY,*/
//*        YOU MUST EDIT THE MEMBERS 'ACTIVITY' AND 'SECURITY' IN     */
//*        YOUR $CD.MAPLIB DATA SET IN THE FOLLOWING MANNER:          */
//*                                                                   */
//*        CHANGE THE SAS STATEMENT ...                               */
//*        PROC PRINTTO NEW UNIT=20;   TO ...                         */
//*                                                                   */
//*        PROC PRINTTO UNIT=20;                                      */
//*                                                                   */
//*        THIS WILL CAUSE THE OUTPUT FROM EACH REPORT TO APPEND TO   */
//*        THE END OF THE OUTPUT FILE RATHER THAN OVERLAY WHAT'S      */
//*        ALREADY THERE.                                             */
//*                                                                   */
//* 10) CHANGE $CD.MAPLIB TO THE NAME OF THE DATA SET CONTAINING      */
//*     THE SAS PROGRAMS.                                             */
//*                                                                   */
//* NOTE - IF YOU WOULD LIKE TO PUT SPECIFIC BOUNDARIES ON THE TIME   */
//* SPAN TO BE REPORTED ON, MODIFY THE SEL STATISTICS COMMAND IN      */
//* THE SYSIN STREAM OF THE DMBATCH STEP TO CONTAIN START AND STOP    */
//* TIMES.  AS IS, REPORTING WILL OCCUR ON THE ENTIRE STAT FILE.      */
//*********************************************************************/
//DMBATCH      EXEC PGM=DMBATCH,PARM='YYSLN'
//STEPLIB      DD    DSN=$CD.LINKLIB,DISP=SHR
//DMPUBLIB     DD    DSN=$CD.PROCESS,DISP=SHR
//DMMSGFIL     DD    DSN=$CD.MSG,DISP=SHR
//TEMPDSN      DD    DSN=&CDAPI,DISP=(NEW,PASS),DCB=(DSORG=PS,
//                   RECFM=VB,LRECL=4100,BLKSIZE=4104),SPACE=(4104,(70,13)),
//                   UNIT=$UNITNAME
//DMPRINT      DD    SYSOUT=*
//SYSPRINT     DD    SYSOUT=*
//SYSIN        DD*
SIGNON USERID=($UID) NETMAP=$CD.NETMAP TMPDD=TEMPDSN
SEL STAT WHERE () FILE
SIGNOFF
/*
//*
///SASTEP EXEC $SASPROC,
//    OPTIONS='DQUOTE MACRO MACROGEN MERROR MISSING="-"'
//WORK         DD    UNIT=$UNITNAME,SPACE=(CYL,(20,10))
//NDMX0001     DD    DISP=(OLD,DELETE),DSN=&CDAPI
//DMMSGFIL     DD    DISP=SHR,DSN=$CD.MSG
//SASPROGS     DD    DISP=SHR,DSN=$CD.MAPLIB
//FT20F001     DD    SYSOUT=*
//*FT20F001    DD    DISP=MOD,DSN=$OUTPUT.DATASET.NAME
//SYSIN        DD *
%INCLUDE SASPROGS(ACTIVITY,SUMMARY,EXCEPT,SECURITY,NPDSCOPY,PDSCOPY,
 RUNJOB,RUNTASK,SUBMIT);
//*
```

Job Stream Definitions

The following table defines the parameters and SAS options for the sample job stream.

Report Field	Definition
JOBCARD INFO	Specifies the typical job card information. Use the COND=(0,NE) for best results. This means that the second step of the job does not run if an error occurs in the first step.
DMBATCH	Specifies the program name of the Connect:Direct batch interface.
STEPLIB	Specifies the library containing the Connect:Direct load modules.
DMPUBLIB	Specifies the library containing the Connect:Direct Processes.
DMMSGFIL	Specifies the Connect:Direct message data set that accesses the Connect:Direct messages.
TEMPDSN	Specifies the temporary data set containing the extract from the Connect:Direct statistics file that is used as input to SAS. This is always specified as (NEW, PASS) so that when you create the data set, the extracted data is saved in it, and then passed on to the next step.
DMPRINT (Optional)	Specifies where the job output from DMBATCH goes. This is useful if an error occurs in DMBATCH.
SYSPRINT	Specifies where the job execution messages goes.
SYSIN	Contains the Connect:Direct control statements to extract from the statistics file. You can represent SYSIN as a sequential data set, PDS member, or instream data set.
SASTEP	Specifies the name of the SAS cataloged procedure used at your installation with the following options: DQUOTE specifies that the system accepts double quotes. MACRO specifies that the SAS macro library is available. MACROGEN specifies that the system can print SAS macros. MERROR specifies that a warning message is produced if a name is prefixed with a percent sign to indicate a SAS macro, but the name is not a valid SAS macro. MISSING specifies that dashes (-) are inserted when a numeric character is missing in the report field.
WORK	Specifies the work area used by SAS for processing.
NDMX0001	Specifies the temporary data set that contains the extract from the Connect:Direct statistics file. Specify the data set parameters (OLD, DELETE) so that the data set is deleted automatically after processing.
DMMSGFIL	Specifies the name of the Connect:Direct message data set.
SASPROGS	Identifies the name of the data set containing the SAS routines to produce the reports. See %INCLUDE parameter in this table.
FT20F001	Specifies the DD statement used by SAS to route output. This DD statement can be either output to a data set or SYSOUT class. If it is to a data set, you must preallocate the data set, it must be sequential, and have the following attributes: RECFM=FBA; LRECL=240; BLKSIZE=3120.

Report Field	Definition
SYSIN	Specifies SYSIN for SAS.
%INCLUDE	Contains a special SAS control statement that enables you to execute SAS programs that are stored separately from their JCL. The first name after SASPROGS identifies the DD name that references the location of the SAS programs. The requested ARS report routine names are placed inside of the parentheses: ACTIVITY specifies the Connect:Direct Activity Report EXCEPT specifies the Connect:Direct PDS Copy Report RUNJOB specifies the Connect:Direct Run Job Report. RUNTASK specifies the Connect:Direct Run Task Report. SECURITY specifies the Connect:Direct Security Violations Report. SUBMIT specifies the Connect:Direct Submit Within a Process Report. SUMMARY specifies the Connect:Direct Summary Report.

Sample Connect:Direct Process That Submits Job Stream

The RPTPROC member in $CD.SAMPLIB in the following figure contains a Connect:Direct Process that runs the job stream discussed in the previous section, *Modifying the Sample Job Stream* on page 42. You can set up the sample Process to run automatically on a specified time interval, by submitting this Process with the Connect:Direct SUBMIT command, specifying the STARTT parameter and RETAIN=YES parameter.

```
RPTPROC                          -
* THIS PROCESS WILL SUBMIT FOR EXECUTION THE JCL TO RUN ARS
* CHANGE $SECONDARY.NODE TO THE DESIRED SNODE NAME FOR THE PROCESS
* CHANGE $TSOID TO YOUR TSO ID
* CHANGE $CD.CNTL(ARSJCL) TO THE DATA SET NAME CONTAINING THE JCL
*        TO RUN ARS
 PROCESS                      -
  SNODE=$SECONDARY.NODE       -
  NOTIFY=$TSOID
 STEPONE                      -
     RUN JOB (                -
      DSN=$CD.CNTL(ARSJCL)    -
            )
```

You need to modify the following items if you choose to use this sample Process.

1. Change $SECONDARY.NODE to the appropriate SNODE.
2. Change $TSOID to your TSO ID.
3. Change $CD.CNTL(ARSJCL) to the data set name containing the job that runs ARS. (See STEPONE.)

For convenience, you may want to replace RPTPROC in your Connect:Direct Process library after editing it.

Routing Multiple Reports to an Output Data Set

Complete this step only when you request multiple ARS report types in one job and the reports are sent to an output data set. The Connect:Direct Activity Report and Connect:Direct Security Violations Report require a minor edit change in the ARS routine if you include them in a multiple report request.

Before modifying the ARS routines, you may want to make a copy of the $CD.MAPLIB, which contains all of the report routines. Use the first copy when you request only one report (in batch mode or using the ARS screens). The second copy can contain the modified routines for use when you make multiple report requests in a job.

You can report the ARS report routines using TSO. The routines are in the $CD.MAPLIB. The following table lists the types of reports and the corresponding members in the $CD.MAPLIB that require modification.

Report Type	Member Name
Connect:Direct Activity Report	ACTIVITY
Connect:Direct Security Violations Report	SECURITY

Modify the PROC PRINTTO statement listed in the members so that the output from one report appends rather than overlaying output from the previous report in the designated output file. Change the ARS report routine from:

```
PROC PRINTTO NEW UNIT=20
```

to:

```
PROC PRINTTO UNIT=20;
```

Chapter 6

ARS Record Layouts

This chapter contains the information you need to customize ARS reports. You can either modify the ARS reports or you can develop new reports. Examples include the following:

- Change ARS report headings, spacing, field titles, and output format
- Access additional information from the Connect:Direct statistics file to enhance ARS reports or to develop new reports

The ARS routines do not utilize all of the Connect:Direct fields in the Connect:Direct statistics records. To assist you in customizing reports, this chapter lists all of the Connect:Direct fields. The SAS Informat variables for these records are located in the $CD.MAPLIB library. Access the information by the member names listed in the following table.

Note: Use the two character designations for records types when you browse the Connect:Direct statistics file.

Member Name	Contents of Member	Record Types
CDAER	Authorization Event Record	IU=INSert USER UU=UPDate USER SU=SELect USER DU=DELete USER
CDCPTR	Change Process Termination Record	CH=CHange PROCess
CDCTR	Copy Termination Record	CT=COPY
CDDPTR	Delete Process Termination Record	DP=DELete PROCess
CDDTR	Display Statistics Record	SP=SELect PROCess DT=SELect TASK FT=FLUSH TASK SS=SELect STATistics SN=SELect NETmap
CDFPTR	Flush Process Termination Record	FP=FLUSH PROCess
CDFMCR	PDS Member Copy Record	MC=PDS member COPY

Member Name	Contents of Member	Record Types
CDPSSR	Process Submit Statistics Record	PS=SUBmit statement SW=SUBmit command
CDPTR	Process Termination Record	PT=PROCess
CDRJTR	Run Job Termination Record	RJ=RUN JOB
CDRTTR	Run Task Termination Record	RT=RUN TASK
CDSFR	Signon/Signoff Statistics Record	SI=SIGNON SO=SIGNOFF
CDSDCR	Start Connect:Direct Command Record	SD=Start Connect:Direct
CDSTDCR	Stop Connect:Direct Statistics Record	ST=STOP CD
CDFWTOST	Write to Operator (WTO) Statistics Record	WT=Write to Operator

Description of an SAS Variable

The SAS Informat variables for these records are located in the $CD.MAPLIB library. An example of an SAS Informat variable is @5 SASFIELD PK1. Variable descriptions for this example are listed in the following table.

Variable	Description
@n	Identifies the position of the variable in the Connect:Direct statistics record (relative to 1). Always identified by an @.
SASFIELD	Specifies the SAS name for the corresponding Connect:Direct field name. The SAS name variable must follow the position variable (@n).
PK1	Specifies the type and length of the SAS Informat variable used in the maps. The type variable must follow the name variable (SASFIELD). Type variable used in the maps are: HEX—numeric hexadecimal IB—integer binary PD—packed decimal PIB—positive integer binary PK—packed unsigned $n—standard character $CHAR—characters with blanks $HEX—character hexadecimal $VARYING—variable-length values

Authorization Event Record

The following table shows each field available in the Connect:Direct Authorization Event record. The CDAER member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
AERECLN	Length of this record.
AERTYPE	Record type indicates specific data in statistics record: IU=INSert; UU=UPDate USER; SU=SELect USER; and DU=DELete USER.
AERTIME	Time that event was recorded in hh/mm/ss/tt format.
AEDATE	Date that event was recorded in Julian date format (yyyydddf).
AEPROCNM	Process name.
AEPROCNO	Process number.
AEUNODE	User node of the submitter.
AEUID	Userid of the submitter.
AESTEP	Step name or label.
AESTIME	Time that Process started in hh/mm/ss/tt format.
AESDATE	Date that Process started in Julian date format (yyyydddf).
AESCC	Step completion code.
AEMSGID	Message ID.
AESDSNL	Length of data set name.
AESDSN	Source data set name.
AESDSTYP	Data set type (LIB, DSN).
AEEVENT	Event code.
AEEVENTQ	Event code qualifier.

Change Process Termination Record

The following table shows each field available in the Connect:Direct Change Process Termination record. The CDCPTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
CHRECLN	Length of this record.
CHRTYPE	Record type – CH indicates the Change Process Termination Record.
CHTIME	Time that CHange PROCess command completed in hh/mm/ss/tt format.
CHDATE	Date that CHange PROCess command completed in Julian date format (yyyydddf).
CHPROCNM	Process name.
CHPROCNO	Process number.
CHSTEP	Step name or label.
CHUNODE	User node ID of the submitter.
CHUID	Userid of the submitter.
CHSTIME	Time that CHange PROCess command started in hh/mm/ss/tt format.
CHSDATE	Date that CHange PROCess command started in Julian date format (yyyydddf).
CHSCC	Step completion code-displays normal completion code.
CHMSGID	Message ID.
CHRMNID	Node where message routed.
CHRMUID	Userid where message routed.
CHSCHTME	Time that Process was scheduled in hh/mm/ss/tt format.
CHSCHDTE	Date that Process was scheduled in Julian date format (yyyydddf).
CHSCHDAY	Day that Process was scheduled.
CHPRTY	New Process priority in the Connect:Direct Transmission Control Queue (TCQ).
CHRETAIN	Keeps copy in TCQ after execution.
CHPROC	ALL/PROCNAME/PROCNUMBER.
CHNDEST	New destination node.
CHHOLD	Process status in hold queue: Q=quiesce or I=immediate.
CHNPRTY	New priority.
CHREL	Release Processes=R.
CHRET	Retain Processes=R.

Field Names	Field Description
CHNRMNID	New node ID messages to be routed.
CHNRMUID	New userid messages to be routed.
CHNEWTME	New scheduled time in hh/mm/ss/tt format.
CHNEWDTE	New scheduled date in Julian date format (yyyydddf).

Copy Termination Record

The following table shows each field available in the Connect:Direct Copy Termination record. The CDCTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	FIeld Description
CTRECLN	Length of this record.
CTRTYPE	Record type – CT indicates the Copy Termination record.
CTTIME	Time that COPY step completed in hh/mm/ss/tt format.
CTDATE	Date that COPY step completed in Julian date format (yyyydddf).
CTSTIME	Time that COPY step started in hh/mm/ss/tt format.
CTSDATE	Date that COPY step started in Julian date format (yyyydddf).
CTSCC	Step completion code. Displays normal completion code.
CTMSGID	Message ID.
CTPROCNM	Process name.
CTPROCNO	Process number.
CTSTEP	Step name or label.
CTUNODE	User node ID of the submitter.
CTUID	Userid of the submitter.
CTPNODE	Name of the node which is examining its Connect:Direct statistics file.
CTSNODE	Name of the other node.
CTNODE	This node is P(node) or S(node).
CTFROM	Direction of data: Snode to Pnode or Pnode to Snode

Field Names	Fleld Description
CTTRANS	File translation. Includes: ASCII-TO-EBCDIC; EBCDIC-TO-ASCII; EBCDIC-TO-ASCII DIF; compressed; and compacted.
CTTRANS2	Transmission options. Includes: PDS-to-PDS copy; error originated on other node; sending PDS member to sequential DSN; sending DSN sequential DSN to PDS member; and COPY is restarted.
CTRUSZ	Request/response unit (RU) size.
CTPACCT#	Displacement to Pnode account data length.
CTSACCT#	Displacement to Snode account data length.
CTINBYTE	Number of bytes read from data set.
CTINRECN	Number of records read from data set.
CTINBLK	Number of blocks read from data set.
CTSBYTES	Number of bytes sent.
CTOBYTE	Number of bytes written to data set.
CTOTRECN	Number of records written to data set.
CTOTBLK	Number of blocks written to data set.
CTRBYTES	Number of bytes received.
CTNOKB	Number of kilobytes sent.
CTNOMEMS	Number of members sent.
CTNOALIS	Number of aliases sent.
CTNOMEMX	Number of members selected but not sent.
CTNOALIX	Number of aliases selected but not sent.
CTNOMEMR	Number of members received.
CTNOALIR	Number of aliases received.
CTCMPTBL	The compaction table name.
CTSDSTYP	Source data set type: ESDS, KSDS, RRDS, PDS, SAM, or LIB.
CTSDISP1	Source data set disposition status: O=OLD and S=SHR.
CTSDISP2	Source data set disposition normal termination: D=DELETE and K=KEEP.
CTSDISP3	Source data set disposition abnormal termination: D=DELETE and K=KEEP.
CTSDSNL	Length of the source data set name.
CTSDSN	Source data set name.
CTTYPE	Type file key, if Type is specified in a Process step.

Field Names	FIeld Description
CTDDSTYP	Destination data set type.
CTDDISP1	Destination data set disposition status: M=MOD; N=NEW; O=OLD; and S=SHR.
CTDDISP2	Destination data set disposition normal termination: C=CATLG and K=KEEP.
CTDDISP3	Destination data set disposition abnormal termination: C=CATLG; K=KEEP; and D=DELETE.
CTDDSNL	Length of destination data set name.
CTDDSN	Destination data set name.
CTMEMBER	Member name for PS to PO copies.
CTOMSGID	Message ID from other node.
CTRTNCD	Return code from other node.
CTNVTAMS	Number of VTAM sends.
CTNVTAMR	Number of VTAM receives.
CTDSPVLR	Displacement of receiving VOLSER list.
CTDSPVLS	Displacement of sending VOLSER list
CTTBOFS	TCP buffer size used.
CTV2BUFS	V2 buffer size.
CTNEGBF	Negotiated V2 buffer size.

Delete Process Termination Record

The following table shows each field available in the Connect:Direct Delete Process Termination record. The CDDPTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
DPRECLN	Length of this record.
DPRTYPE	Record type—DP indicates the Delete Process Termination record.
DPTIME	Time that DELete PROCess command completed in hh/mm/ss/tt format.

Field Names	Field Description
DPDATE	Date that DELete PROCess command completed in Julian date format (yyyydddf).
DPPROCNM	Process name.
DPPROCNO	Process number.
DPSTEP	Step name or label.
DPUNODE	User node iD of the submitter.
DPUID	Userid of the submitter.
DPSTIME	Time that DELete PROCess command started in hh/mm/ss/tt format.
DPSDATE	Date that DELete PROCess command started in Julian format (yyyydddf).
DPSCC	Step completion code. Displays normal completion code.
DPMSGID	Message ID.
DPDPRNM	Name of the deleted Process.
DPDPRNOP	Number of the deleted Process.
DPRMNID	Route message node of deleted Process.
DPRMUID	Route message userid of deleted Process.

Display Statistics Record

The following table contains each field available in the Connect:Direct Display Statistics record. The CDDTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Name	Field Description
DTRECLN	Length of this record.
DTRTYPE	Record type indicates specific data in statistics record: SP=SELect PROCess; DT=SELect TASK; FT=FLUSH TASK; SS=SELect STATistics; and SN=SELect NETMAP.
DTTIME	Time that command completed in hh/mm/ss/tt format.
DTDATE	Date that command completed in Julian date format (yyyydddf).
DTPROCNM	Process name.
DTPROCNO	Process number.

Field Name	Field Description
DTSTEP	Step name or label.
DTUNODE	User node ID of the submitter.
DTUID	Userid of the submitter.
DTSTIME	Time that command started in hh/mm/ss/tt format.
DTSDATE	Date that command started in Julian date format (yyyydddf).
DTSCC	Step completion code. Displays normal completion code.
DTMSGID	Message ID.
DTDFLNM	Name of the displayed data set.
DTNVTAMS	Number of VTAM sends.
DTNVTAMR	Number of VTAM receives.
DTGETS	Number of GETS.
DTPUTS	Number of PUTS.

Flush/Suspend Process Termination Record

The following table shows each field available in the Connect:Direct Flush Process/Suspend Process Termination record. The CDFPTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
FPRECLN	Length of this record.
FPRTYPE	Record type–FP indicates the Flush Process Termination record.
FPTIME	Time that FLUSH PROCess command completed in hh/mm/ss/tt format.
FPDATE	Date that FLUSH PROCess command completed in Julian date format (yyyydddf).
FPPROCNM	Process name.
FPPROCNO	Process number.
FPSTEP	Step name or label.
FPUNODE	User node ID of the submitter.
FPUID	Userid of the submitter.

Field Names	Field Description
FPSTIME	Time that FLUSH PROCess command started in hh/mm/ss/tt format.
FPSDATE	Date that FLUSH PROCess command started in Julian date format (yyyydddf).
FPSCC	Step completion code. Displays normal completion code.
FPMSGID	Message ID.
FPDPRNM	Name of the flushed Process.
FPDPRNOP	Number of the flushed Process.
FPRMNID	Route message node of flushed Process.
FPRMUID	Route message userid of flushed Process.

PDS Member Copy Record

The following table shows each field available in the Connect:Direct PDS Member Copy record. The CDFMCR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
MCRECLN	Length of this record.
MCRTYPE	Record type–MC indicates the PDS Member Copy record.
MCTIME	Time that COPY step completed in hh/mm/ss/tt format.
MCDATE	Date that COPY step completed in Julian date format (yyyydddf).
MCSTIME	Time that member processing started in hh/mm/ss/tt format.
MCSDATE	Date that member processing started in Julian date format (yyyydddf).
MCSCC	Step completion code. Displays normal completion code.
MCMSGID	Message ID.
MCPROCNM	Process name.
MCPROCNO	Process number.
MCSTEP	Step name or label.
MCUNODE	User node ID of the submitter.
MCUID	Userid of the submitter.

Field Names	Field Description
MCPNODE	The Pnode name for this Process.
MCSNODE	The Snode name for this Process.
MCNODE	This node is P(node) or S(node).
MCFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
MCTNAME	Name of member on destination PDS.
MCFNAME	Name of member on source PDS (if different).
MCANAME	Name of member for which node is an alias.
MCNRECS	The number of records.
MCNBLKS	The number of blocks.

Process Submit Statistics Record

The following table shows each field available in the Connect:Direct Process Submit Statistics record. The CDPSSR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
PSSRECLN	Length of this record.
PSSRTYPE	Record type indicates specific data in statistics record: PS=SUBmit statement and SW=SUBmit command within a Process.
PSSCTIME	Time that Process completed in hh/mm/ss/tt format.
PSSCDATE	Date that Process completed in Julian date format (yyyydddf).
PSSSTIME	Time that Process started in hh/mm/ss/tt format.
PSSSDATE	Date that Process started in Julian date format (yyyydddf).
PSSSCC	Step completion code. Displays normal completion code.
PSSMSGID	Message ID.
PSSPRCNM	Process name.
PSSPRCNO	Process number.
PSSSTEP	Step name or label.
PSSUNODE	Submitter’s symbolic node name.

Field Names	Field Description
PSSUID	Userid of the submitter.
PSSPNODE	Pnode name for this Process.
PSSSNODE	Snode name for this Process.
PSSNODE	This node is P(node) or S(node).
PSSFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
PSSROSIZ	The number of bytes in the Process.
PSSRMNID	Route message userid of Process.
PSSFUNCD	The function code for the SUBMIT.
PSSSYNTX	The syntax error in SUBmit command.
PSSPARSE	The parse error in SUBmit command.
PSSPACT#	Displacement to Pnode account data.
PSSSACT#	Displacement to Snode account data.
PSSDSN#	Displacement to data set name.

Process Termination Record

The following table shows each field available in the Connect:Direct Process Termination record. The CDPTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
PTRECLN	Length of this record.
PTRTYPE	Record type–PT indicates the Process Termination record.
PTTIME	Time that Process completed in hh/mm/ss/tt format.
PTDATE	Date that Process completed in Julian date format (yyyydddf).
PTSTIME	Time that Process started in hh/mm/ss/tt format.
PTSDATE	Date that Process started in Julian date format (yyyydddf)
PTSCC	Step completion code. Displays normal completion code.
PTMSGID	Message ID.
PTPROCNM	Process name.

Field Names	Field Description
PTPROCNO	Process number.
PTSTEP	Step name or label.
PTUNODE	Submitter's symbolic node name.
PTUID	Userid of the submitter.
PTPNODE	Pnode name for this Process.
PTSNODE	Snode name for this Process.
PTNODE	This node is P(node) or S(node).
PTFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
PTUNPRNO	Process number unique to submitter's node.
PTRMNID	Route message userid of Process.
PTRMUID	Userid where message is routed.
PTSUBTME	Time that Process was submitted in hh/mm/ss/tt format.
PTSUBDTE	Date that Process submitted in Julian date format (yyyydddf).
PTSCHTME	Time that Process was scheduled in hh/mm/ss/tt format.
PTSCHDTE	Date that Process was scheduled in Julian date format (yyyydddf).
PTSCHDAY	Day Process was scheduled.
PTSTMTN	Number of statements.
PTPRTY	Process selection priority.

Run Job Termination Record

The following table shows each field available in the Connect:Direct Run Job Termination record. The CDRJTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
RJRECLN	Length of this record.
RJRTYPE	Record type–RJ indicates the Run Job Termination record.
RJTIME	Time that RUN JOB step completed in hh/mm/ss/tt format.
RJDATE	Date that RUN JOB step completed in Julian date format (yyyydddf).

Field Names	Field Description
RJSTIME	Time that RUN JOB step started in hh/mm/ss/tt format.
RJSDATE	Date that RUN JOB step started in Julian date format (yyyydddf).
RJSCC	Step completion code. Displays normal completion code.
RJMSGID	Message ID.
RJPROCNM	Process name.
RJPROCNO	Process number.
RJSTEP	Step name or label.
RJUNODE	Submitter's symbolic node name.
RJUID	Userid of the submitter.
RJPNODE	Pnode name for this Process.
RJSNODE	Snode name for this Process.
RJNODE	This node is P(node) or S(node).
RJFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
RJJOBNM	Job name.
RJJOBNO	Job number.
RJSYRCD	System return code.
RJDDSN	Displacement to data set name.
RJPACCT#	Displacement to Pnode account data.
RJSACCT#	Displacement to Snode account data.

Run Task Termination Record

The following table shows each field available in the Connect:Direct Run Task Termination record. The CDRTTR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
RTRECLN	Length of this record.
RTRTYPE	Record type–RT indicates the Run Task Termination record.
RTTIME	Time that RUN TASK step completed in hh/mm/ss/tt format.

Field Names	Field Description
RTDATE	Date that RUN TASK step completed in Julian date format (yyyydddf).
RTSTIME	Time that RUN TASK step started in hh/mm/ss/tt format.
RTSDATE	Date that RUN TASK step started in Julian date format (yyyydddf).
RTSCC	Step completion code. Displays normal completion code.
RTMSGID	Message ID.
RTPROCNM	Process name.
RTPROCNO	Process number.
RTSTEP	Step name or label.
RTUNODE	Submitter's symbolic node name.
RTUID	Userid of the submitter.
RTPNODE	Pnode name for this Process.
RTSNODE	Snode name for this Process.
RTNODE	This node is P(node) or S(node).
RTFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
RTMODNM	Name of the RUN TASK program.
RTSYSRTN	System return code.
RTPARMLN	Length of the parameter list.
RTPACCT#	Displacement to Pnode account data.
RTSACCT#	Displacement to Snode account data.

Signon/Signoff Statistics Record

The following table shows each field available in the Connect:Direct Signon/Signoff Statistics record. The CDSFR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
SIRECLN	Length of this record.
SIRTYPE	Record type indicates specific data in the statistics record: SI=SIGNON command and SO=SIGNOFF command.

Field Names	Field Description
SITIME	Time that SIGNON/SIGNOFF command completed in hh/mm/ss/tt format.
SIDATE	Date that SIGNON/SIGNOFF command completed in Julian date format (yyyydddf).
SIUNODE	User symbolic node ID.
SIUID	Userid of the submitter.
SISCC	Step completion code. Displays normal completion code.
SIMSGID	Message ID.

Start Connect:Direct Command Record

The following table shows each field available in the Start Connect:Direct Command record. The CDSDCR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
SDRECLN	Length of this record.
SDRTYPE	Record type–SD indicates the Start Connect:Direct Command record.
SDTIME	Time that Start Connect:Direct command completed in hh/mm/ss/tt format.
SDDATE	Date that Start Connect:Direct command completed in Julian date format (yyyydddf).
SDPROCNM	Process name.
SDPROCNO	Process number.
SDSTEP	Step name or label.
SDUNODE	Submitter's symbolic node name.
SDUID	Userid of the submitter.
SDSTIME	Time that Start Connect:Direct command was issued in hh/mm/ss/tt format.
SDSDATE	Date that Start Connect:Direct command was issued in Julian date format (yyyydddf).
SDSCC	Step completion code. Displays normal completion code.
SDMSGID	Message ID.

Stop Connect:Direct Statistics Record

The following table shows each field available in the Stop Connect:Direct Statistics record. The CDSTDCR member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
TDRECLN	Length of this record.
TDRTYPE	Record type–ST indicates the Stop Connect:Direct Statistics record.
TDTIME	Time that STOP Connect:Direct command completed in hh/mm/ss/tt format.
TDDATE	Date that STOP Connect:Direct command completed in Julian date format (yyyydddf).
TDPROCNM	Process name.
TDPROCNO	Process number.
TDSTEP	Step name or label.
TDUNODE	User node of the submitter.
TDUID	Userid of the submitter.
TDSTIME	Time that STOP Connect:Direct command was issued in hh/mm/ss/tt format.
TDSDATE	Date that STOP Connect:Direct command was issued in Julian date format (yyyydddf).
TDSCC	Step completion code. Displays normal completion code.
TDSTOP	Connect:Direct shutdown types: W=Step shutdown; X=Immediate shutdown; Y=Quiescent shutdown; and Z=Forced shutdown by an abend.
TDMSGID	Message ID.

Write to Operator (WTO) Statistics Record

The following table shows each field available in the Connect:Direct WTO Statistics record. The CDFWTOST member of the $CD.MAPLIB library contains the SAS maps of the Connect:Direct statistics record.

Field Names	Field Description
WTRECLN	Length of this record.
WTRTYPE	Record type–WT indicates the WTO Statistics record.
WTCTIME	Time that Process completed in hh/mm/ss/tt format.
WTCDATE	Date that Process completed in Julian date format (yyyydddf).
WTSTIME	Time that function started in hh/mm/ss/tt format.
WTSDATE	Date that function started in Julian date format (yyyydddf).
WTSCC	Step completion code. Displays normal completion code.
WTMSGID	Message ID.
WTPROCNM	Process name.
WTPROCNO	Process number.
WTSTEP	Step name or label.
WTUNODE	Submitter’s symbolic node name.
WTUID	Userid of the submitter.
WTPNODE	The Pnode name for this Process.
WTSNODE	The Snode name for this Process.
WTNODE	This node is P(node) or S(node).
WTFROM	Direction of data: P(node) to Snode or S(node) to Pnode.
WTTASKNO	Task number that issued the WTO.

Chapter 7

Connect:Direct for z/OS Operator Interface

The Operator Interface enables you to issue all Connect:Direct for z/OS commands from a z/OS console by using a MODIFY command. The command results are displayed on the console.

This chapter explains how to:

- Invoke the Operator Interface
- Use the Operator Interface
- Interpret Connect:Direct operation error messages
- Stop Connect:Direct
- Conduct tape-handling procedures for Connect:Direct for z/OS

Invoking the Connect:Direct for z/OS Operator Interface

You can access the Operator Interface after Connect:Direct initializes if you specify the Connect:Direct initialization parameters MCS.CLIST and MCS.SIGNON.

The syntax for the MCS.CLIST and MCS.SIGNON initialization parameters follows.

```
MCS.CLIST=console operator's CLIST library (dsn)
MCS.SIGNON=(SIGNON USERID=(userid,password)        -
      CASE=YES | NO-
           NETMAP=network map file name)
```

An automatic signon is issued after the first Connect:Direct CLIST is invoked using the information in MCS.SIGNON. This session is active until you submit a CLIST that contains a Connect:Direct SIGNOFF command.

Using the Operator Interface

You can create easy to remember customized Connect:Direct commands with the Operator Interface through a CLIST-type facility. The software supports symbolic substitution and CLIST-type parameters allowing you to alter Connect:Direct commands without changing the CLIST.

In a Connect:Direct/Plex, if you issue a console operator command to a Connect:Direct/Server, the console interface actually signs on to the Connect:Direct/Manager. As a result, any commands issued to Connect:Direct/Server are actually issued to the Connect:Direct/Manager, which is the only Connect:Direct/Plex member that accepts operator commands.

Sample Connect:Direct for z/OS CLISTs

The Connect:Direct sample CLIST library, OPLIST on the distribution tape, contains the sample CLISTs included in the following table. Use these CLISTs to build customized Connect:Direct commands with symbolic parameters that allow you to customize the CLIST at submission time. Sample operator commands for each CLIST are listed in the comment section.

CLIST Name	Command	Description
CMD		Provides a general format for requesting a Connect:Direct command.
DP	DELETE PROCESS	Shows six ways to delete a Process using this CLIST.
DUMP	STOP CD (Force)	Stops Connect:Direct using the FORCE parameter and produces abnormal ending (abend) 4095 and a dump.
IPRM	DISPLAY INITPARMS	Displays initialization parameters to the operator console. In a PLEX environment, it works as follows: ◆ Displays the global initialization parameters ◆ Defaults to the manager's local initialization parameters ◆ Allows the name of a server to display the server's global and local initialization parameters
MSG	SELECT MSG	Shows a detail of the Connect:Direct message requested.
NM	SELECT NETMAP	Displays entire NETMAP or it allows for one parameter – a node name.
RLSE	CHANGE PROCESS	Shows three ways to release a Process.
SIGNOFF	SIGNOFF	Signs the operator off Connect:Direct.
SIGNON	SIGNON	Connects the operator to Connect:Direct.
SP	SELECT PROCESS	Shows six ways to display Process status information.

CLIST Name	Command	Description
SS	SELECT STATISTICS	Shows seven ways to select statistics for display.
STATS	SELECT STATISTICS	Shows three ways to select statistics for display covering a specific time period.
STOP	STOP CD	Stops Connect:Direct using the STOP IMMEDIATE command. To perform an immediate shutdown in a PLEX environment from the operator console, issue the following command: F CD,STOP | CDPLEX. To stop a specific server, use this command: F CD,STOP | WHERE(SERVER=name)
SUB	SUBMIT PROCESS	Shows two ways to submit a Process.
SUSPEND	SUSPEND PROCESS	Shows four ways to suspend a Process.
SWAP	SWAP NODE	Swaps operator to another node in a multiple session environment.
TS	SELECT TASK	Shows active tasks.
VP	VIEW PROCESS	Displays a specific process.

Rules for Setting Up Connect:Direct for z/OS CLISTs

The following rules apply when setting up an operator CLIST:

- All operator CLISTs must have a PROC record as the first record in the CLIST. The PROC record defines the parameters and keywords that are passed to the CLIST.
- You can stack multiple Connect:Direct commands in one CLIST, but you only need one PROC statement.
- A number between **0** and **24** (inclusive) must follow the PROC identifier. This number indicates the number of positional parameters used by the CLIST.
- Parameter names (one to eight characters each) that correspond to each positional parameter follow the number. If no positional parameters exist, specify PROC 0.
- If you define more than one positional parameter on the PROC statement, you do not need to specify trailing positional parameters in the command unless you also specify a keyword parameter.
- Use commas to indicate null values.
- A positional parameter is terminated by the first blank encountered.
- You can specify optional keyword parameters after the positional parameters. Enclose default values in parentheses after each keyword name.
- Use a hyphen (-) to indicate that a Connect:Direct command continues on the next line. An example follows.

```
PROC 1 PNUM
SELECT PROCESS WHERE( -
         PNUM=&PNUM)
```

- You can use comments in the CLIST only if you include an asterisk (*) in the first column.
- Sequence numbers are not allowed.

Submitting Connect:Direct for z/OS Commands

Submit the Connect:Direct commands to the Operator Interface using a Connect:Direct for z/OS MODIFY command interface. This operator command lets you submit a CLIST containing Connect:Direct commands, and modify the Connect:Direct commands by substituting symbolic parameters with real values.

The command format is:

```
F jobname,clist [options]
```

The following table describes the parameters in the preceding command format:

Parameter	Description
F	The short form of the Connect:Direct for z/OS MODIFY command.
jobname	The name on the job statement of the job stream that brings up the DTF. **Caution**: If jobname is a Connect:Direct/Server, the command is redirected to the Connect:Direct/Manager.
clist	The name of the CLIST containing the Connect:Direct command.
options	Optional positional and keyword parameters to be passed to the CLIST.

Examples

The following are CLIST examples and operator commands that submit the CLIST examples. As some examples show, you can use symbolic parameters to modify the Connect:Direct command at the time of submission. For example purposes, CDDTF is the jobname.

CLIST with Command and No Parameters

In this example, the CLIST SELX contains one command and no parameters.

```
PROC 0
SEL PROC WHERE(QUEUE=EXEC)
```

To execute SELX, type the following operator command.

```
F CDDTF,SELX
```

When you issue the operator command, a Select Process command for Processes on the executing queue initiates. As a result, a list of these Processes is displayed.

CLIST with Command and One Parameter

In this example, the CLIST SELQ contains one command and one positional parameter (SUBD).

```
PROC 1 SUBD
SEL PROC WHERE(QUEUE=&SUBD)
```

The following options present two ways to submit SELQ:

- When you issue the following command, TIMER (timer queue) is substituted for &SUBD. This command displays all Processes on the timer queue.

```
F CDDTF,SELQ TIMER
```

- When you issue the following command, ALL is substituted for &SUBD in the CLIST SELQ. This command displays all Processes on all queues.

```
F CDDTF,SELQ ALL
```

CLIST with Command and Multiple Parameters

In this example, the CLIST SUB contains one command, two positional parameters (PROC and PARMS), and one keyword parameter (SNODE).

```
PROC 2 PROC PARMS SNODE(OTHER.NODE)
SUB PROC=&PROC SNODE=&SNODE &PARMS
```

The following options present four ways to execute SUB:

- When you issue the following command, PAYROLL is substituted for &PROC. This substitution causes the Process PAYROLL to be submitted.

```
F CDDTF,SUB PAYROLL
```

  Because no SNODE is specified, the SNODE defaults to OTHER.NODE.

- When you issue the following command, CHECKS is substituted for &PROC and HOLD=YES is substituted for &PARMS. These cause the Process CHECKS to be submitted as a held Process.

```
F CDDTF,SUB CHECKS HOLD=YES
```

  Because no SNODE is specified, the SNODE defaults to OTHER.NODE.

- When you issue this command, ORDERS is substituted for &PROC and CD.CHICAGO is substituted for &SNODE. These cause the Process ORDERS to be submitted to run in session with the secondary node called CD.CHICAGO.

```
F CDDTF,SUB ORDERS,,SNODE=(CD.CHICAGO)
```

  Because PROC and PARMS are positional parameters, commas are required.

- When you issue the following command, CONFIRM is substituted for &PROC and HOLD=NO,STARTT=(FRIDAY,NOON) is substituted for &PARMS. These submit the Process CONFIRM so that it runs at noon on Friday.

```
F CDDTF,SUB CONFIRM HOLD=NO,STARTT=(FRIDAY,NOON)
```

  The SNODE defaults to OTHER.NODE because no SNODE is specified.

CLIST with Command and %IF %ELSE %EIF

In this example, the CLIST NM contains 1 command and 1 positional parameter.

```
PROC 1 NODE
    %IF &NODE = ,
  SEL NETMAP WHERE (NODE=*)
    %ELSE
  SEL NETMAP WHERE (NODE=&NODE)
    %EIF
```

If you enter no operands it will display the entire netmap. I you enter a particular node it will display just that node.

Interpreting Connect:Direct for z/OS Operation Messages

Connect:Direct operation diagnostic messages are formatted into multiple lines so that errors and exception conditions are easy to read. Operator messages are sent to:

- The route code given in the Connect:Direct initialization parameters
- The ddname NDMLOG, if it is allocated

The COPY termination message (SVTM052I) is displayed in four lines, as follows:

- The first line shows the step label, the operation (COPY), the Process name and number, the associated node name, and the session class.
- The second line shows the FROM file name.
- The third line shows the TO file name.
- The fourth line shows the completion code and Connect:Direct message ID. If the COPY did not complete successfully, the first four positions of the last line contain the "####" flag.

When session errors occur, Connect:Direct formats the error message into two lines, as follows:

- The first line gives as much information as possible in the same format as the COPY termination message (Process name and number, and associated node name).
- The second line gives the text of the error message. The first four positions of the second line contain the "****" flag.

The following example shows two Connect:Direct operation error messages following a COPY termination message.

```
SVTM052I STEP1  COPY  PROCESS1( 126)PNODE=CD.NODE.A  (002)
   FROM  SYSA.SAM.DATA.SET
   TO    SYSB.TEST.SAM.DATA.SET
   #### COMPLETED 00000010/SCPA010I
SVTM045I              PROCESS1( 126)PNODE=CD.NODE.A
   **** RPLERRCK:CD/CD SESSION FAILURE
SVTM050I              PROCESS1( 126)PNODE=CD.NODE.A
   **** PROCESS INTERRUPTED:RECOVERY INITIATED
```

Stopping Connect:Direct

To stop Connect:Direct for z/OS from the console, use the following command, which executes the STOP CLIST using the Operator Interface.

```
F jobname,STOP
```

Stopping a Connect:Direct/Plex Manager and All Servers

To stop a Connect:Direct/Plex from the console, use the following command which executes the STOP CLIST using the Operator Interface:

```
F jobname,STOP [Q|S|I|R|F,] CDPLEX [,RECOVER]
```

Stopping an Individual Connect:Direct/Plex Server

To stop a Connect:Direct/Plex Server from the console, use the following command which executes the STOP CLIST using the Operator Interface:

```
F jobname,STOP [Q|S|I|R|F,] WHERE(SERVER=name) [,RECOVER]
```

For more information, please refer to the Stopping Connect:Direct topic in the *Connect:Direct for z/OS Administration Guide*. For information on restarting Connect:Direct, see the *Connect:Direct for z/OS Installation Guide*.

Understanding Connect:Direct for z/OS Tape Mount Messages

Connect:Direct for z/OS supports the transmission of tape files. Connect:Direct implements separate and distinct messages to control tape management, providing the greatest flexibility to operators in managing tape transfers. The following sections describe the three Connect:Direct tape messages.

Tape Pre-mount Message

This optional message is issued prior to allocating a tape device. It tells you that Connect:Direct requires a specific volume on a specific device type. With this message, you can:

- Control the number of tape units available to Connect:Direct for tape transfers.
- Locate the volumes before the transfer occurs.

You are required to respond to this message when the required resources are available.

Tape Mount Messages

These messages are also optional. The first message (WTO) is in the format of the z/OS mount message. The second message is issued as a WTOR to inform the operator to mount a specific volume on a specific device. The third message is issued only if TAPE.PREMOUNT=LIST. Message SVST00C is displayed listing all volume-serial numbers of the requested file. If you are unable to satisfy the specific request, you can reply CANCEL to cancel the request. This message is also used to drive the visual display devices attached to 3480-type devices.

Note: If you use tape silos in your environment, see the information on the Tapemount Exit in the chapter on using Connect:Direct exits in the *Connect:Direct for z/OS Administration Guide*. To prevent the Connect:Direct environment from locking up due to an outstanding tape mount request, the Connect:Direct tapemount exit provides an interface to StorageTek Tape Silo Software to query the silo to determine that all VOLS required for a tape file are present.

Setting Up Connect:Direct for z/OS Tape Pre-mount Messages

To issue the Connect:Direct tape pre-mount messages, specify TAPE.PREMOUNT=YES|LIST in the Connect:Direct initialization parameter file. See the *Connect:Direct for z/OS Installation Guide* for details.

Setting Up Connect:Direct for z/OS Tape Mount Messages

Issue the Connect:Direct tape mount messages in place of the z/OS mount message. The Connect:Direct mount messages consist of a z/OS-format mount message followed by a WTOR issued to the descriptor and route codes specified by the DESC.TAPE and ROUTCDE.TAPE.

You can suppress the Connect:Direct message by specifying ROUTCDE.TAPE=(0) in the initialization parameters. The defaults are DESC.TAPE=(2) ROUTCDE.TAPE=(5,11). If you suppress Connect:Direct tape mount messages, the standard z/OS mount messages are issued with related serialization during mount processing.

If you specify ROUTCDE.TAPE=(0), causing Connect:Direct not to issue the mount messages, then the normal z/OS mount message is issued when the dataset is opened. The z/OS mount processing holds an ENQ on the SYSZTIOT resource. This ENQ causes all other Connect:Direct Processes to hang at allocation, open, and deallocation until the tape is mounted.

If you use the Connect:Direct tape mount message, the tape already is on the tape drive at open time and z/OS mount processing does not hold the SYSZTIOT resource.

Responding to Connect:Direct for z/OS Tape Pre-mount Messages

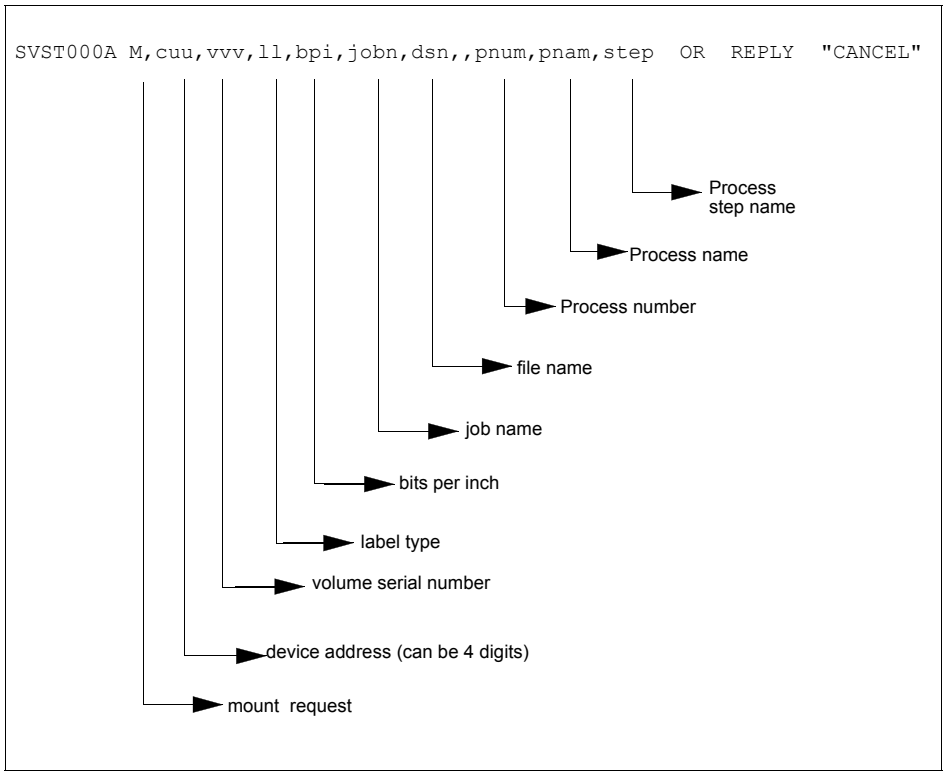
The following is the format of a Connect:Direct tape pre-mount message.

```
SVST000B - C:D REQUIRES VOL=SCRTCH TYPE=TAPE 6250/1600BPI
REPLY "GO" WHEN READY
```

To continue processing, reply GO to the message. While this message is outstanding, no unit is allocated. However, the session for this transfer request remains active.

Responding to Connect:Direct for z/OS Tape Mount Messages

The following is the format of a Connect:Direct tape mount message issued during a specific tape request.



If no tape is available or if you want to end the copy, reply CANCEL to the tape mount message.

Tape Device Allocation

After the GO reply to a pre-mount message, and following the pre-mount processing, Connect:Direct allocates a tape device using Defer mounting. If an allocation error occurs, and the

allocation error code is in the ALLOC.CODES parameter of the Connect:Direct initialization parameters, Connect:Direct retries the operation.

When device allocation completes successfully, Connect:Direct issues a z/OS-format Open mount message to allow tape management and visual display routines to process the request. The format of the message follows.

```
IEC501A M,cuu,volser,sl,,jobn,dsn,pnum,pnam,step
```

Following the IEC501A message, Connect:Direct issues a WTOR to allow you to mount the tape or cancel the request.

Terminating the Tape Mount

A COPY step within a Process requiring a tape to be mounted causes the tape mount message to display. You can terminate the COPY step with a CANCEL reply. Any other reply causes Connect:Direct to reissue the mount message.

If you are using both the tape pre-mount message and the tape mount message, and want to cancel the request, reply GO to the tape pre-mount (SVST000B) and CANCEL to the tape mount message (SVST000A). If you do not want to terminate the copy step, the mount message disappears when the tape is mounted.

Verifying Volume Requests

For standard label tapes, Connect:Direct for z/OS verifies the volume serial number for a specific volume request. If you mount the wrong volume, the following message is issued and the volume is dismounted.

```
SVST001I - D cuu,volser --- IS NOT THE REQUESTED VOLUME
```

You can mount the correct volume, or you can reply CANCEL to the reissued SVST000A mount message.

Handling Multivolume Files

The Connect:Direct for z/OS mount message is issued only for the first volume of a file, when a multivolume file is requested. Subsequent mount requests causes a SYSZTIOT enqueue during end-of-volume processing.

To keep Connect:Direct from long waits while a second or subsequent volume is mounted, use UNIT=(unit,P) or UNIT=(unit,n) for the UNIT keyword of the COPY statement. For a description of the COPY statement, see the Connect:Direct Processes Web site at http://www.sterlingcommerce.com/Documentation/processes/processhome.html.

Connect:Direct for z/OS Messages on 3480 Display

The 3480 tape drive has an 8-character display. Connect:Direct issues messages to this display concerning tape processing in the following manner:

- When a tape is to be dismounted, the first character position of the display contains a **D**. Positions 2–7 consist of the tape's volume serial number. Position eight contains an **N** if the tape is nonlabeled or an **S** if the tape has standard labels.
- When a tape is to be mounted, the first character position of the display contains an **M**. Positions 2–7 consist of the tape's volume serial number. Position eight contains an **N** if the tape is nonlabeled or an **S** if the tape has standard labels.
- When a tape is loaded, the first character position of the display is blank. Positions 2–7 consist of the tape's volume serial number. Position eight contains an **N** if the tape is nonlabeled or an **S** if the tape has standard labels.

Chapter 8

Event Services Support

Connect:Direct Event Services Support (ESS) implements an asynchronous, event-generation facility in Connect:Direct. ESS is designed for use by external management and automated operations applications that require real-time notification of Connect:Direct activities. ESS supports a publish or subscribe protocol with optional guaranteed event delivery and automated recovery.

Customer applications can register interest in receiving event data, and specify the types of event data the application receives through the new event services commands. ESS supports a general user exit point and an optional API for interface communication with user-written CICS automation applications.

Concepts and Components

ESS provides real-time delivery of Connect:Direct event records to interested application interfaces. ESS is based on the existing Connect:Direct statistics facility and is designed to:

- Minimize the server-performance overhead for ESS processing
- Prevent adverse impact on Connect:Direct server reliability

ESS is a low-level Connect:Direct server technology that enables you to layer multiple APIs. An end-user application interfaces with the API rather than the low-level ESS core technology. This design enables future extension of ESS to support multiple application interface protocols and simultaneous publishing of the same event data to several event consumer applications.

System Interfaces

ESS supports the event services exit and the CICS API interface.

An application can access event data through an ESS exit. A sample event exit (ESSEVX01) is in the sample library, and writes the event data to a predefined data set.

A CICS API initiates event processing by signing on to Connect:Direct through the current CICS API and issuing ESS commands. The CICS API provided with Connect:Direct writes event data to a CICS Transient Data Queue (TDQ). A sample program, DMQ249, shows how to read an event record from the TDQ.

System Advantages

Building on the existing statistics facility has the following advantages:

- Event Definition

  Because ESS event records are Connect:Direct statistics records, ESS does not restrict events to abnormal activities. This design enables ESS-enabled applications to initiate procedures based on either successful or unsuccessful Process statements.

- Historical Data Recovery

  The Connect:Direct statistics file functions as the historical repository of ESS event data. ESS supports optional automated recovery of past event data if an event consumer application fails. Upon restart, ESS requests old event records from the statistics file.

  The API automatically manages the restart point for an ESS-enabled application. This API tracks the last successfully delivered event record to a designated subscriber. Upon restart, the API resynchronizes event processing by issuing a SELect STATistics command to retrieve the historical event data and deliver it to the restarted event consumer.

- Event Record Types

  The event record types give external applications greater visibility into Connect:Direct activities and improve the reliability and responsiveness of interface applications in a production environment. ESS supports several Connect:Direct event record types, including:

  - **Process Initialization (PI)** occurs immediately prior to the execution of the first Process statement in a Process.
  - **Step Start (CI, JI, TI)** events occur immediately prior to the execution of a COPY, RUN JOB, or RUN TASK statement.
  - **COPY Step I/O Start (CE)** occurs immediately prior to the first COPY step I/O in a Process.
  - **Queue Change (QW, QH, QE, QT)** events occur whenever a Process moves from one logical queue to another in the TCQ. The sample initialization parameters provided in the base Connect:Direct install the PARMLIB data set and specify STAT.EXCLUDE=(QE, QH, QT, QW). This specification prevents Connect:Direct from producing the queue change records. If you want queue change records, you must set this parameter appropriately.

  See Chapter 12, *Event Services Record Descriptions*, for more information about Event Record Types.

ESS also supports basic event filtering. When an event consumer application registers interest in receiving events, the application can restrict the types of event data that it wants to receive. These options support the same filtering syntax as the SELect STATistics command. Following are two possible selections:

- A fault management application can request notification of COPY requests that did not complete normally, QH (Queue Hold) events or both. When such an event occurs, the fault management application can then poll Connect:Direct through SELect STATistics or SELect PROCess to investigate the problem further.
- An application can register interest in all events for a small group of mission-critical Processes.

Chapter 9

Using Event Services Support

ESS uses three main components to provide real-time information about Connect:Direct activities:

- Client application
- API
- Connect:Direct server

ESS can communicate with client application interfaces through the CICS API or through an ESS user exit.

The ESS architecture described in this chapter provides an overview of the system structure. Event data flows are presented in subsequent diagrams. A summary of each type of interface follows the discussion of system operation.

Connect:Direct Event Services Support Architecture

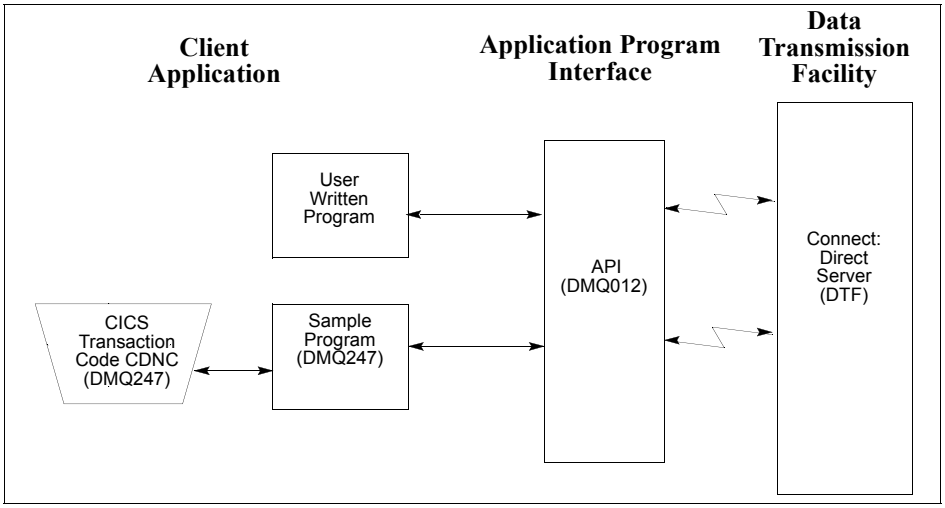
The three main components of ESS interact to accomplish the following tasks:

- Define which events are reported to the client application
- Turn event services requests on and off
- Register event data
- Report event data

The following diagram details ESS with a CICS application using the Connect:Direct CICS API. The structure consists of the following components:

- The client application defines the event data requests and issues commands through a user-written program. The DMQ247 sample program demonstrates the method to issue the ESS CDNC commands.
- The API receives and passes event data and ESS CDNC commands to the Connect:Direct server, and maintains information for the guaranteed delivery feature.

- The Connect:Direct server is the Data Transmission Facility (DTF) that processes ESS CDNC commands and sends the requested event data to the client application through the API (DMQ012).



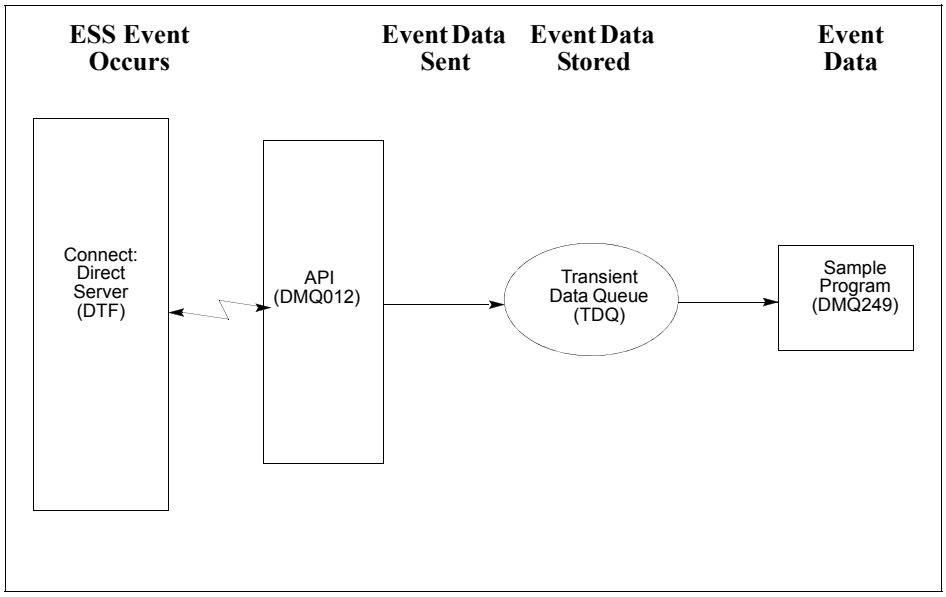
ESS Data Flow Using CICS API

You define the event services environment by specifying the types of event data your application receives. These definitions are the event data requests that you issue through the DMQ247 program or a user-written program.

After you define the event data requests for your client application and activate your request by issuing the EVENT SERVICES START command, ESS processes the requested event data as follows:

- Upon receiving the EVENT SERVICES START ID parameter, the API (DMQ012) passes the command to the DTF through a VTAM session.
- When a requested event occurs, the DTF sends the event data to the API for distribution to the Transient Data Queue (TDQ).
- The event data is then available to the client application through the DMQ249 Sample Program. The DMQ249 program provides a way to read event data from the TDQ and processes the data for use by your application.

The following diagram illustrates the flow of event data from the DTF to the client application:



For additional information on Event Services commands, refer to Chapter 10, *Issuing Event Services Commands*.

Deciding What Event Data to Collect

Deciding what event data to collect requires that you understand the types of data the ESS delivers and how you can use this data in your event consumer application. Although it is technically possible to request all event data for all Processes in the system, it is usually unnecessary and impractical to do so. Typically, an event consumer application is interested in a subset of event data.

Note: Statistics records excluded through the API STAT command or a DTF statistics exit, or by coding the DTF STAT.EXCLUDE initialization parameter, are not available to ESS.

The following are commonly requested event data:

- All event data for a particular Process or select group of Processes

  Requesting all event data gives the most detailed trace of Process execution. This information is useful during testing of an event consumer application. However, it is typically unnecessary after such an application goes into production.

- Major event data for a particular Process or select group of Processes

  This request is the most common subset specification. For normal production work, this request gives your application a sufficient trace of Process execution, and minimizes overhead for ESS processing and the processing of event records by your application.

- Subset of event data for a particular Process, set of Processes, or all Processes

  For example, an application executes a procedure when a Connect:Direct Process terminates. In this case, it may be sufficient to register interest in PT (Process Termination) records only. As a second example, a fault reporting application needs CT (Copy Termination) statements that did not complete normally (signalling the failure of a Connect:Direct file transfer request).

ESS Event Record Examples

The following examples show the ESS event records from a single Copy Process. The examples show the records produced from Process submittal until normal Process termination. The examples assume that no Connect:Direct commands affecting the Process [including the SELect PROCess command] are issued between Process submittal and termination.

- ESS Event Records Example 1 (No Retries)

  In this example, the Process does not undergo a Process retry for any reason. The sequence of ESS event records produced by this Process are listed in the following table. These records are in sequential order.

Record Type	Record Description
PS	Process submit
QE	Process moved to EXEC queue
PI	Process initiation
CI	COPY step statement start
CE	COPY step I/O start
CT	COPY termination
NL	Last statement (null step 1)
PT	Process termination

- ESS Event Records Example 2 (One Retry)

  This example shows the sequence of ESS records that are produced if the same Process undergoes a single Process retry. Connect:Direct commands are deliberately issued to suspend the Process and then release it for retry. As a result, the following record sequence includes the various operator commands required. The events are in sequential order.

Record Type	Record Description
PS	Process submit
QE	Process moved to EXEC queue
PI	Process initiation

Record Type	Record Description
CI	COPY step statement start
CE	COPY step I/O start
FS	Suspend Process command issued
CT	COPY termination (as a result of requested SUSPEND)
QH	Process moved to HOLD queue
CH	Change Process command issued with RELEASE=Yes
QE	Process moved to EXEC queue
PI	Process initiation
CI	COPY step start
CE	COPY I/O start
CT	COPY termination
NL	Last statement in Process end
PT	Process termination

Using ESS with the CICS API

The CICS API writes event data to the Transient Data Queue (TDQ). Your CICS application must include a program to read these event records from the TDQ and pass them to your application for processing.

Follow these steps to enable a CICS application for ESS:

1. Decide what event data your application requires. Refer to *Deciding What Event Data to Collect* on page 83 for additional information.
2. Verify that the Connect:Direct CICS API software is correctly installed and working. Refer to the Connect:Direct installation documentation for further information.
3. Create an ESS registration program. Include this program in the startup portion of your application. It must be called whenever your application wants to initiate event record collection. This program must perform the following:
   - Sign on to the Connect:Direct CICS API.
   - Issue the EVENT SERVICES CREATE command to register interest in receiving event data and to specify event record filter criteria.

- Specify the ID parameter to identify a subscriber name for your application. Specify the WHERE and ORWHERE parameters for your filtering criteria. Refer to *EVENT SERVICES CREATE Command Format* on page 89 for further information about event filter criteria.
- Issue the EVENT SERVICES START command and specify the ID parameter to initiate event reception.

4. Create a customer interface program to read event records off the CICS TDQ and pass them to your application for processing. The sample interface program (DMQ249) demonstrates how to read an event record from the TDQ. Specifically, your application must issue an EXEC CICS READQ TD QUEUE instruction.
5. Modify your application to process the ESS event records.
6. Embed a call to the EVENT SERVICES STOP ID=subscriber_name parameter in your application where you want to stop collecting event data.

Using ESS with the ESS User Exit

You can execute Event Services commands through any existing Connect:Direct API such as CICS, ISPF, and DMBATCH. However, not all Connect:Direct APIs can process synchronous event data received from an EVENT SERVICES START command. In some cases, it is preferable not to have the Connect:Direct API process synchronous event data. The event services exit provides an alternative. Rather than passing synchronous event data back to the API that issued the EVENT SERVICES START (ESS) command, an exit processes the event data.

System Architecture

The ESS exit acts as an extension to the Connect:Direct server. When a synchronous event occurs, Connect:Direct calls the event services exit. An event exit control block is passed for each event and contains the following:

- Address of the event record
- Flags indicating first or last call
- Fields for the user to indicate return code and message

DMEVEXCB, found in the sample library, maps the event exit control block. The sample exit, ESSEVX01, is also in the sample library.

The event services exit is called as a subtask from Connect:Direct so that the user code waiting for system services does not directly effect Connect:Direct processing. The sample exit (ESSEVX01) writes each event record to a predefined data set. You must modify ESSEVX01 to specify the name of your event exit data set, and you must define the data set to accommodate records up to 2048 bytes in length.

Using the ESS Exit

Each record passed to the exit consists of a record header and an event record. DMMNHDR, which is in sample library, maps the record header. Members in the sample library also map the event records. See Chapter 12, *Event Services Record Descriptions,* to determine which member maps a particular event record.

The exit must reside in a load library accessible to the Connect:Direct server (DTF). Connect:Direct moves event data to storage below the 16M line before calling the event services exit. This step accommodates systems where I/O calls must be done below the line.

To use the ESS user exit, follow these steps:

1. Create an ESS exit program that can process the requested event records.
2. Decide what event data your application requires. Refer to *Deciding What Event Data to Collect* on page 83 for additional information.
3. Verify that your software is correctly installed and working.
4. Create an ESS registration program. Include this program in the startup portion of your application. It is called whenever your application wants to initiate event record collection. This program must accomplish the following:
   - Signon to Connect:Direct through the CICS API, the ISPF IUI, or DMBATCH.
   - Issue the EVENT SERVICES CREATE command to register interest in receiving event data and specify event record filter criteria.

     Specify the EXIT parameter to identify the name of your exit program, and the ID parameter to identify a subscriber name for your application. Specify the WHERE and ORWHERE parameters for your filtering criteria. Refer to *EVENT SERVICES CREATE Command Format* on page 89 for more information about event filter criteria.
   - Issue the EVENT SERVICES START command and specify the ID parameter to initiate event reception.
5. Embed a call to the EVENT SERVICES STOP ID=subscriber_name parameter in your application where you want to stop collecting event data.

If the user application does not process records fast enough, the internal queue, which holds asynchronous events, becomes full. When the queue is full, event processing for this request ends, and Connect:Direct writes a WTOR message. To customize the internal queue size, specify the MAXQCNT parameter on the EVENT SERVICES CREATE command.

Chapter 10

Issuing Event Services Commands

The Event Services commands are the tools that enable you to determine what information your application receives, and to start and stop event notification. You can define the event data through an event services request. This chapter details how to use the EVENT SERVICES commands to create requests, start and stop event notification, and display event data.

This chapter describes the four commands that the Connect:Direct Event Services Support uses:

- EVENT SERVICES CREATE
- EVENT SERVICES START
- EVENT SERVICES STOP
- EVENT SERVICES DISPLAY

Each command description includes the function of the command and the command parameters.

EVENT SERVICES CREATE Command Format

The EVENT SERVICES CREATE command enables you to create a new request for event data and define which data is sent to your application.

The EVENT SERVICES CREATE command uses the following format and parameters:

Command	Parameters
EVENT [SERVICES] CREATE	**ID=event-request-name**
	WHERE(CCODE|PNAME|PNUMber|STARTT|STOPT|USER|SNODE|TYPE| FNAME|CASE|LASTSEQ|MAXQCNT|EXIT)
	ORWHERE|=(CCODE|PNAME|PNUMber|USER|SNODE|TYPE| FNAME)

Required Parameters

The EVENT SERVICES CREATE command has two required parameters.

ID=event-request-name

specifies a logical name for the event services request. This name is a text string of 1–16 characters. You can use any printable characters. You cannot use embedded blanks.

WHERE (CCODE = (condition, completion code)
PNAME = name | (list)
PNUMber = number | (list)
STARTT = ([date | day] [,hh:mm:ssXM])
STOPT = ([date | day] [,hh:mm:ssXM])
USER = name | (list)
SNODE = name | (list)
TYPE = id | (list)
FNAME = dsname | (list)
CASE = YES | NO
LASTSEQ=n
MXQCNT=n
EXIT=exitname)

specifies selection criteria for event records.

You must specify at least one WHERE () subparameter, such as CCODE, PNAME, PNUMBER, or TYPE.

CCODE = (condition, completion code)

specifies selection by completion code.

condition specifies a relationship to the completion code given in the subsequent positional parameter. The options for specifying condition are:

GT greater than
LT less than
EQ equal to
NE not equal to
GE greater than or equal to
LE less than or equal to

completion specifies a completion code value ranging from 1 to 2,147,483,647 so the RUN TASK can pass all values.

For example, if you specify CCODE = (GT,0), you see event records in which the step completion code is greater than zero, as long as the records also meet other specified criteria.

PNAME = name | (list)

specifies selection by Process name. Specify a list of Processes by enclosing them in parentheses. You can use a wild card character (*) at the end of the name.

For example, if you specify PNAME=TEST*, then all records with TEST in the first four characters of the Process name field are selected. Records having TEST, TEST123, and TESTX all satisfy the selection criterion.

name specifies the name of the Process to select.

(list) specifies a list of Process names to select. Enclose the list in parentheses. Separate Processes in the list with commas.

PNUMber = number | (list)

specifies selection by Process number. To request a list of Processes, enclose them in parentheses. The range is 1–99999.

number specifies the number of the Process to select.

(list) specifies a list of Process numbers to select. Enclose the list in parentheses. Separate Processes in the list with commas.

STARTT = ([date | day] [,hh:mm:ssXM])

specifies selection by start date and time. Specify STARTT as a date and time prior to the current time. If you set this parameter, the event records retrieved from the statistics file begin with the date and time you specified and continue until the current time is reached. At that point, event records are processed as they occur.

Note: If you specify STARTT and LASTSEQ with the EVENT SERVICES START command, those values replace the values specified for these parameters in the EVENT SERVICES CREATE command. When you use the time specification in conjunction with the LASTSEQ parameter, you must include hours, minutes, seconds, and hundredths of seconds in the format hh:mm:ss.th.

---

date specifies the starting date from which event records are retrieved from the statistics file. You can specify the date in either Gregorian or Julian format.

If you use a Gregorian date format, set the DATEFORM initialization parameter to the appropriate date format. Otherwise, the date format defaults to the platform date format. You do not need to set the DATEFORM parameter for Julian date format.

The following table shows the acceptable date formats.

Format	DATEFORM Parameter	Date Format
Gregorian	DATEFORM=MDY	mmddyy or mmddyyyy mm/dd/yy or mm/dd/yyyy mm.dd.yy or mm.dd.yyyy
Gregorian	DATEFORM=DMY	ddmmyy or ddmmyyyy dd/mm/yy or dd/mm/yyyy dd.mm.yy or dd.mm.yyyy
Gregorian	DATEFORM=YMD	yymmdd or yyyymmdd yy/mm/dd or yyyy/mm/dd yy.mm.dd or yyyy.mm.dd
Gregorian	DATEFORM=YDM	yyddmm or yyyyddmm yy/dd/mm or yyyy/dd/mm yy.dd.mm or yyyy.dd.mm
Julian	N/A	yyddd or yyyyddd yy/ddd or yyyy/ddd yy.ddd or yyyy.ddd

Note: If RETAIN=Y, you cannot specify a date in the STARTT parameter of the initialization file.

day specifies the day of the week to select. Valid names include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. You can also specify YESTER to retrieve event records created since yesterday, or TODAY to retrieve event records created today.

hh:mm:ssXM indicates the start time of day in hours (hh), minutes (mm), and seconds (ss) selected. XM indicates AM or PM. You can use the 24-hour clock or the 12-hour clock. If you use the 24-hour clock, valid times are 00:00-24:00. If you use the 12-hour clock, valid times are 00:00-12:00 and you must indicate AM or PM. For example, 01:00 hours on the 24 hour clock is expressed as 1:00AM on the 12 hour clock. If you do not specify AM or PM, Connect:Direct assumes the 24-hour clock. You do not need to specify minutes and seconds. You can also specify NOON to retrieve event records starting at noon, or MIDNIGHT to retrieve event records starting at midnight. The default for the time is 00:00:00, the beginning of the day.

If you do not specify the STARTT parameter, event processing begins with the current time.

STOPT = ([date | day] [,hh:mm:ssXM])

specifies when interest in event data ends.

date specifies the stop date when event processing ends. You can specify the date in either Gregorian or Julian format.

Refer to the STARTT subparameter on page 91 for a discussion of Gregorian and Julian date formats.

If you specify only the date, the time defaults to 24:00:00.

day specifies the day of the week to select. Valid values include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. You can also specify TODAY.

hh:mm:ssXM indicates the stop time of day in hours (hh), minutes (mm), and seconds (ss) to select. XM indicates AM or PM. You can use the 24-hour clock or the 12-hour clock. If you use the 24-hour clock, valid times are 00:00-24:00. If you use the 12-hour clock, valid times are 00:00-12:00 and you must indicate AM or PM. For example, 01:00 hours on the 24 hour clock is expressed as 1:00AM on the 12 hour clock. If you do not specify AM or PM, Connect:Direct assumes the 24-hour clock.

You do not need to specify minutes and seconds. You can also specify NOON to retrieve event records starting at noon, or MIDNIGHT to retrieve event records starting at midnight. The default time is 24:00:00, the end of the day.

If you do not specify the STOPT parameter, you must issue an EVENT SERVICES STOP command to stop event processing.

USER = name | (list)

limits the selected event records to those written for userids with the specified name. You can specify a list of names by enclosing them in parentheses. You can use wild card characters. For example, if you specify USER = SYS$*, then records with SYS$ in the first four characters of the userid field are selected. Records having SYSBOB, SYSADM, and SYS$0001 all satisfy this selection criterion. Userid names can be up to 64 characters in length and can contain lowercase characters.

name specifies the userid to select.

(list) specifies a list of userids to select. Enclose the list in parentheses. Separate userids in the list with commas.

SNODE = name | (list)

limits the selected event records to those written for Processes where the specified node name acted as SNODE. You can specify a list of names by enclosing them in parentheses. You can use wild card characters. For example, if you specify SNODE=DALLAS*, then all records with DALLAS in the first six characters of the SNODE field are selected. Records having DALLAS.PROD, DALLAS.TEST, and DALLAS all satisfy this selection criterion. SNODE names can contain lowercase characters.

name specifies the SNODE to select.

(list) specifies a list of SNODES to select. Enclose the list in parentheses. Separate SNODES in the list with commas.

TYPE = id | (list)

specifies the event record types to select. Every event record that Connect:Direct generates has an associated record-type identifier. Each identifier is two characters long and indicates the event or function that generated the record. The identifier also indicates the record format and contents.

id specifies the event record type.

(list) specifies a list of event record types to select. Enclose the list in parentheses. Separate record types in the list with commas.

See Chapter 12, *Event Services Record Descriptions*, for a list of all event record type identifiers.

FNAME=dsname | (list)

limits the selected event records to those that contain the specified filename. The FNAME subparameter is valid for the following record types: Copy Termination (CT), Run Job (RJ), Start Connect:Direct (SD), and Submit within Process (SW).

name specifies the filename to select.

(list) specifies a list of filenames to select. Enclose the list in parentheses. Separate filenames in the list with commas.

The meaning of the filename within these records is unique for each record type. For example, the Run Job record contains the filename of the submitted JCL. Filenames can be up to 254 characters in length and can contain lowercase characters. Filenames must conform to your platform's naming conventions.

CASE=YES | NO

specifies whether lowercase or mixed-case data is permitted for the USER, SNODE, and FNAME subparameters. The CASE subparameter overrides the global CASE option defined at signon for the SELECT STATISTICS command.

YES changes the data in USER, SNODE, and FNAME to uppercase regardless of the actual data specified.

NO preserves the actual case entered for the USER, SNODE, and FNAME subparameters.

The CASE defaults to the setting defined in the session defaults, if nothing is specified.

LASTSEQ=n

specifies the last sequence number (for Restart capability) associated with the date/time stamp. Use it with the STARTT parameter. This information is provided by CICS when you issue a CREATE EVENT SERVICES REQUEST command for an event services request that finished abnormally. For additional information about how to use LASTSEQ, refer to *Using LASTSEQ* on page 103.

MAXQCNT=n

specifies the maximum number of elements that can reside in the internal event queue for this request. All events that match the WHERE criteria for this request are placed on an internal queue. If events occur faster than the API can process them, the queue can become full. If the queue becomes full, Connect:Direct terminates event services requests. You can

modify the queue size using MAXQCNT. Each queue element takes approximately 2K in storage. The default is **100**.

EXIT=exitname

specifies the name of an EVENT SERVICES OPTION exit. The sample exit ESSEVX01 is provided in the sample library. Event data is processed by an exit rather than the calling API, if EXIT is specified. Refer to *Using ESS with the ESS User Exit* on page 86 for information about the user exit.

Optional Parameters

The following parameters are optional for the EVENT SERVICES CREATE command.

ORWHERE(CCODE = (condition, completion code
PNAME = name | (list)
PNUMber = number | (list)
USER = name | (list)
SNODE = name | (list)
TYPE = id| (list)
FNAME = dsname | (list))

specifies which statistics records you want to examine.

The subparameters, such as CCODE, PNAME, PNUMBER, and TYPE, are optional, but you must specify at least one.

The parameters for the ORWHERE keyword are defined in the WHERE keyword section beginning on page 89.

Sample Command

The following is an example of the EVENT SERVICES CREATE command.

```
EVENT CREATE ID=PROCESS_TERM WHERE(TYPE=PT)
```

EVENT SERVICES START Command Format

The EVENT SERVICES START command enables you to start a specific notification request for event data. This command provides parameters where you can define what data (date, time, and sequence) is sent to your application.

The EVENT SERVICES START command uses the following format and parameters.

Command	Parameters
EVENT [SERVICES] START	ID=event-request-name
	WHERE=(STARTT|LASTSEQ| KEEPRREC| (for CICS only) TRACE) (CICS only)

Required Parameter

The EVENT SERVICES START command has one required parameter.

ID=event-request-name

specifies the logical name of the event services request to start. Specify this name in a CREATE EVENT SERVICES command.

Optional Parameters

The following parameter is optional for the EVENT SERVICES START command.

WHERE(STARTT|LASTSEQ|KEEPRREC|TRACE)

sets restriction on event records meeting the following selection criteria.

STARTT=([date|day][,hh:mm:ssXM]) specifies selection by designated starting date and time. STARTT must be a date and time prior to the current time. If you specify this parameter, the event records retrieved from the statistics file begins with the date and time you specified, and continues until the current time is reached. At that point, event records are processed as they occur.

For a complete explanation of the subparameters, refer to *EVENT SERVICES CREATE Command Format* on page 89.

LASTSEQ=n specifies the last sequence number (for Restart capability) associated with the date/time stamp. Use this parameter with the STARTT parameter. This information is provided by CICS when you issue a CREATE EVENT SERVICES REQUEST command for an event services request that finished abnormally. For additional information about how to use LASTSEQ, refer to *Using LASTSEQ* on page 103.

KEEPRREC (CICS-only) specifies that the restart record created by Connect:Direct is kept, regardless of whether the event request completed abnormally or not. If KEEPRREC is not specified, the restart record created by Connect:Direct is deleted if the event request completes normally. Refer to *Using LASTSEQ* on page 103 for more information about how to use the LASTSEQ parameter.

TRACE (CICS-only) specifies that every event record placed on the CICS TDQ is also written to the NDMTRACE temporary storage (TS) queue. You can use the NDMTRACE temporary storage queue to verify that EVENT records are flowing to the transient data queue.

Sample Command

The following is an example of the EVENT SERVICES START command.

```
EVENT START ID=PROCESS_TERM
```

EVENT SERVICES STOP Command Format

The EVENT SERVICES STOP command enables you to stop event notification for a specific event services request.

Note: The EVENT SERVICES STOP command stops event notification and deletes the event request from the system.

The EVENT SERVICES STOP command uses the following format and parameters.

Command	Parameters
EVENT [SERVICES] STOP	ID=event-request-name
	IMMEDIATE

Required Parameter

The EVENT SERVICES STOP command has one required parameter.

ID=event-request-name

specifies the logical name of the event services request to start. This name must match a name specified in a CREATE EVENT SERVICES command.

Optional Parameter

The EVENT SERVICES STOP command has one optional parameter.

IMMEDIATE

indicates that no more event records can be sent. If you do not specify IMMEDIATE, all records on the internal EVENT queue are sent before the EVENT request is terminated.

EVENT SERVICES DISPLAY Command Format

The EVENT SERVICES DISPLAY command enables you to display event data for a specific event services request. The EVENT SERVICES DISPLAY command uses the following format and parameter.

Command	Parameter
EVENT [SERVICES] DISPLAY	ID=event-request-name

Required Parameters

The EVENT SERVICES DISPLAY command has no required parameters.

Optional Parameter

The EVENT SERVICES DISPLAY command has one optional parameter.

ID=event-request-name

specifies the logical name of the event services request to display. This name must match a name specified in a CREATE EVENT SERVICES command. If you do not specify this parameter, all event services requests are displayed.

Chapter 11

Using ESS with the CICS API

The Connect:Direct CICS interface enables you to use Connect:Direct through the Customer Information Control System (CICS) from local and remote sites. The system includes a set of nested menus, prompts for required information, online Help facilities, and monitoring features for current status.

Configuring CICS

In addition to the Connect:Direct IUI, Sterling Commerce provides a facility that enables you to issue standard Connect:Direct commands from a CICS application program. You can use this interface for both terminal and non-terminal tasks. Following are two typical uses for this API:

- For terminal tasks, the API enables an installer to provide their own user interface to all or part of Connect:Direct or work with Connect:Direct from application programs using the CDNC transaction.
- For non-terminal tasks, the API enables an installer to write background transactions which programmatically issue Connect:Direct commands. The most typical sequence is SIGNON - SUBMIT - SIGNOFF. In this case, the SIGNON generates a SIGNON TYPE=CICS, and the SIGNOFF results in a cleanup operation of the signon table entry, temporary storage, and so forth.

  To avoid having to sign on before each command and sign off after each command, the user application can pass a logical task number through the Q012TASK field. If Q012TASK is used, a user program can sign on once and issue multiple commands in pseudo-conversational mode. A signoff is only needed when the user application is terminating.

An example of the use of this facility is provided in member DMQ247 of the $CD.SAMPLIB. This program enables you to type Connect:Direct commands on a screen and view the resulting return code, message number, message text, and Process number assigned to your Connect:Direct Process. In addition, the CICS application displays the name of the CICS Temporary Storage (TS) queue where the results of your command are stored, the count of items in the queue, and the maximum record length in the queue.

You must install both Connect:Direct and the Connect:Direct CICS interface, and they must be operating for the Connect:Direct CICS API to function.

The following components are required to use DMQ247.

Component	Description
DMQ247	The program source, written in assembly language
DMQM98	The BMS map used by DMQ247
DMQBMSTB	A parsing macro used by DMQ247
DMQCA012	A command-level COMMAREA passed to the CICS Interface

A sample Connect:Direct API DRIVER screen follows. This screen shows data after the ESS registration program has signed on to the Connect:Direct CICS API driever and issued an EVENT SERVICES CREATE command.

```
                          CONNECT:Direct API DRIVER


COMMAND         EVENT SERVICES CREATE ID=MESOO031 WHERE(MAXQCNT=998 LASTSEQ=1
                STARTT=(02.14.2005,04:14:12.45) STOPT=(12.31.2041,24:00:00.00)
                TYPE=(QE,QW,QH,QT,PI,CE,CT,PT,CI,RJ,RT,JI,TI,SW,DP) )

COMMAND RC      0000
COMMAND MSG ID SOPM000I
COMMAND MSG     An EVENT REQUEST HAS COMPLETED NORMALLY
PROCESS NUMBER
TD EXIT COUNT
TS QUEUE NAME
TS MAX LRECL

EVENT RESTART DATA
WHERE(STARTT=(00000,00:00:00.00),LASTSEQ=000)




PF keys:  3 Exit   5 Signon   6 Command   7 Signoff
```

Driver Fields

The following table describes the Driver fields:

Field	Description
COMMAND	This 3-line field contains your API command.
COMMAND RC	This 4-character field contains the return code from your API Process.

Field	Description
COMMAND MSG ID	This 8-character field contains the identification number of the message associated with your API Process.
COMMAND MSG	This 64-character field contains the text of the message.
PROCESS NUMBER	This 6-character field contains the Process number assigned by the system to your API Process.
TD EXIT COUNT	This 6-character field contains the number of bytes indicating how much data is written by the exit module for your API Process.
TS QUEUE NAME	This 8-character field contains the name of the TS queue used during your API request.
TS MAX LRECL	This 4-character field contains the maximum logical record length in bytes of the TS queue.
EVENT RESTART DATA	This 62-character field contains the DATE/TIME/SEQ of the last event record successfully received. Data is only displayed if an EVENT SERVICES CREATE command is issued and a previous event services request had ended abnormally.

Using the CICS API Option

To use the sample program, you must first change DMQ247 to issue a valid SIGNON command for your environment (the SIGNON command is defined near the end of the source module). Include a valid USERID, PASSWORD, and NODE. Before executing this program as a transaction, you must be signed on to CICS. Also observe the following items.

- Assemble DMQM98 and then DMQ247. Sample JCL is in the SAMPLIB. Assemble DMQM98 and DMQ247 as follows:
  - Use member ASMBMS to assemble DMQM98
  - Use member ASMCICS to assemble DMQ247
- Use the Connect:Direct Administrative (CDA) transaction to verify that the CICS Interface and the appropriate node are both active.
- Use transaction CDNC to get DMQM98 to display.
- The program checks for the presence of a communications area in the Exec Interface Block (EIB). If none is present, or if you press the **Clear** key, the DMQM98 map is sent and a RETURN TRANSID is performed to invoke the transaction again when you press **Enter**.
- If you pressed the **PF3** or **PF15** key, the program terminates.
- The DMQM98 map is received. If you pressed **PF5** or **PF17**, a Connect:Direct SIGNON request is generated and the results of the command are presented in the map.

- Following a successful signon (that is, the return code on DMQM98 after the SIGNON is zero), you can type a valid Connect:Direct command on the line provided in the DMQM98 map and press **PF6** or **PF18** to send the command to the Connect:Direct DTF. Command results are displayed when they are returned from the DTF.
- To signoff from the DTF, press **PF7** or **PF19**.

When you type a command through the DMQM98 screen, its length is determined and the address of the length and command are placed in the DMQCA012 communications area at label Q012CMDA. Program DMQ012 is then invoked through an EXEC CICS LINK command. When control returns to DMQ247, the DMQCA012 communications area contain the results of the command.

Note: The DMQ247 does not actually display the results of the issued command that are stored in CICS temporary storage. You can retrieve these records programmatically or view them using the CICS CEBR transaction.

Use the techniques in the DMQ247 sample program to issue any valid Connect:Direct command. Results of commands such as SELECT PROCESS and SELECT STATISTICS are written to CICS temporary storage; other commands can produce no output.

Linking DMQ012

Access the API by linking program DMQ012 as follows.

```
EXEC  CICS LINK PROGRAM('DMQ012') COMMAREA(Q012COMM)          +
               LENGTH(Q012CMLH)
```

The COMMAREA Q012COMM is defined by macro DMQCA012 and is also provided in the sample library. A COBOL version of this record layout is provided as member DMQAPIC in the sample library. A brief description of each of the fields follows:

Field	Assembler Directive	Format	Description
Q012CMDA	DS	XL4	Full word containing the address of the command to be issued to Connect:Direct. The command must be in the following format: CMDLEN DS H Length of CMDTEXT that Connect:Direct recognizes, not including length of CMDLEN. CMDTEXT DS Clx Command text

Field	Assembler Directive	Format	Description
Q012RETC	DS	XL4	Return code received after the command is invoked. 0: request completed successfully 1: C:D-CICS interface level error 2: C:D-CICS node level error 3: C:D-CICS signon level error 5: invalid COMMAREA or command passed to CICS API 8 and above: unsuccessful request
Q012TDCT	DS	XL4	TDEXIT ITEM COUNT. The field contains the number of records received from the Connect:Direct DTF.
Q012TDMX	DS	XL2	TDEXIT MAX ITEM SIZE - the maximum record size received from the Connect:Direct DTF.
Q012PROC	DS	CL6	Process number of the latest submitted Process.
Q012TSKY	DS	CL8	Temporary storage ID - the name of the CICS TS queue that contains the command output.
Q012MSID	DS	CL8	Message ID returned from the Connect:Direct for z/OS DTF.
Q012MSTX	DS	CL64	Message Text returned from the Connect:Direct for z/OS DTF.
Q012DTE	DS	PL4	Date of the last event acknowledgment.
Q012TME	DS	XL4	Time of the last event acknowledgment.
Q012SEQ	DS	XL2	Sequence number of the last event acknowledgment.

Q012CMDA is the only required parameter you must set before linking to DMQ012. For optimum performance, follow the example of DMQ247 and initialize all fields before linking to DMQ012.

Any output generated by issuing the command is returned in a temporary storage queue. Your terminal ID is displayed as the last four characters in the unique TS queue name.

Note: DMQ012 requires that it run in CICS key. For the transaction that executes a program linking DMQ012, set the following parameters in the CEDA definitions (or in the RDO for that program):

TaskDataLoc:ANY

TaskDataKey:CICS

Using LASTSEQ

All event records contain a unique timestamp that consists of date, time, and sequence number. Each EVENT SERVICES request is associated with an ID. The Connect:Direct CICS API creates a restart record for an ID when an EVENT SERVICES CREATE command is processed. After an

event record is successfully delivered to the Transient Data Queue, its timestamp is saved in the restart file. If event services processing ends abnormally, the restart record is retained. The restart record is also retained when you specify the KEEPRREC (keep restart record) in the EVENT START statement.

When an EVENT SERVICES CREATE command is processed, the restart file is searched for a record matching the ID of the CREATE. If a match is found, the date, time, and sequence number of the last successfully delivered event record is returned. When you use the time specification in conjunction with the LASTSEQ parameter, you must include hours, minutes, seconds, and hundredths of seconds in the format hh:mm:ss.th.

If missed event data is required, you can use this information as the STARTT and LASTSEQ values as in the following example.

```
EVENT SERVICES START ID=id
WHERE(STARTT=(date,time) LASTSEQ=seq)
```

Event records beginning after this timestamp are processed.

Reading an Event Record from the Transient Data Queue

The sample program DMQ249 reads an ESS event record from the Transient Data queue. The sample program DMQ249 is in $CD.SAMPLIB. You can use sample JCL ASMCICS to assemble DMQ249.

Each record passed to the user application consists of a record header and an event record. DMMNHDR, which is in the Connect:Direct sample library, maps the record header. Members in the sample library also map the event records. See Chapter 12, *Event Services Record Descriptions*, to determine which member maps a particular event record.

Chapter 12

Event Services Record Descriptions

ESS enables you to create ESS-enabled applications. This chapter provides reference information about event record types and attributes that help you create applications. For a list of statistics record types, refer to Chapter 10, *Using Connect:Direct Exits,* in the *Connect:Direct for z/OS Administration Guide*.

Assembler DSECT members are in the sample library. Access the information by the member names listed in the following table. Use the two-character designations for record types when you browse the Connect:Direct statistics file.

Event Record Type Attributes

ESS has several statistics record types. This section summarizes the attributes of these record types.

Most record types have the following common attributes:

- Process name
- Process number
- Message ID
- Submitter's symbolic node name
- PNODE name for this Process
- SNODE name for this Process
- THISNODE (This node is P[node] or S[node].)

The following paragraphs detail the attributes of specific record types. For ease of reference, the information is grouped according to event type or enhancement type, and they are presented alphabetically.

Event Record Types

Record type attributes for event records are described in the following paragraphs. The information is organized alphabetically by record type identifier.

CI (COPY Step Initiation)

This record is created immediately prior to the execution of any tasks associated with a COPY statement. The intent of this record is to capture the timing of such work as data set allocation prior to any I/O occurring. In addition to the common attributes, CI has the following individual attributes:

- COPY step start date/time
- Step name or label
- CI record retry count
- Transfer direction
- Source file name
- Destination file name
- Member name (when applicable)

CE (COPY I/O Start)

This record is created immediately prior to the first block of data transferred in a COPY statement. In addition to the common attributes, CE has the following individual attributes:

- COPY I/O start date/time
- Step name or label
- CE record retry count

EI (Event Request Initiation)

This record is created upon completion of a EVENT SERVICES START command.

Note: The common attributes for Process-related records do not apply to this record.

The following attributes apply to this event:

- Date/Time the command completed
- User ID of the user who issued the command
- Event services request ID
- Completion code for the command
- Command parameter string

EL (Event Session Lost)

This record is created when the Connect:Direct DTF is lost.

ET (Event Request Stop)

This record is created when an EVENT SERVICES STOP command completes.

Note: The common attributes for Process-related records do not apply to this record.

The following attributes apply to this event:

- Date/Time the command completed
- User ID of user who issued the command
- Event services request ID
- Completion code for the command

JI (RUN JOB Start)

This record is created immediately before the job specified by the RUN TASK statement is submitted. In addition to the common attributes, JI has the following individual attributes:

- RUN JOB start date/time
- Step name or label
- Job name
- JI record retry count

QE (Process moved to EXEC queue)

This record is created when a Process is moved to the execution queue. In addition to the common attributes, QE has the following individual attributes:

- Process execution start date/time
- QE record retry count

PI (Process Initiation)

This record is created immediately prior to when the first Process step is executed. In addition to the common attributes, PI has the following individual attributes:

- Queue change date/time
- Execution queue status value
- PI record retry count

QH (Process moved to the HOLD queue)

This record is created whenever a Process is moved to the Hold queue, either through some problem during Process execution or submission, a SUSPEND PROCESS command issued or a session failure. The reason for moving to the Hold queue is generally identified by the Hold queue status value. In addition to the common attributes, QH has the following individual attributes:

- Queue change date/time
- Hold queue status value
- QH record retry count

QT (Process moved to the TIMER queue)

This record is created whenever a Process is moved to the Timer queue, normally through a Process retry. In addition to the common attributes, QT has the following individual attributes:

- Queue change date/time
- Timer queue status value
- QT record retry count

QW (Process moved to the WAIT queue)

This record is created whenever a Process is moved to the Wait queue from the Hold or Timer queues. In addition to the common attributes, QW has the following individual attributes:

- Queue change date/time
- Wait queue status value
- QW record retry count

TI (RUN TASK Initiation)

This record is created immediately prior to the initiation of the task specified by the RUN TASK statement. In addition to the common attributes, TI has the following individual attributes:

- RUN TASK start date/time
- Step name or label
- Program name
- TI record retry count
- Date/time
- Process name and number
- Event name
- Event trigger
- Event type

Event Record Enhancements

An enhancement to existing statistics records is described in the following paragraph.

CT (COPY Termination) Record Enhancements

The existing CT record is enhanced to include the following information:

- Total number of retries for the COPY step
- Total amount of data moved for all attempts of the COPY
- Type keywords specified in a COPY step to retrieve data set defaults for allocating the destination file

Chapter 13

Spool Transfer Facility

Connect:Direct Spool Transfer is an interface that enables you to transfer Job Entry Subsystem (JES) spool files in the following ways:

- Copy from JES Spool Files

  The VPS/CDI option and the VPSSCDI program manage input from JES spool files. If a JES spool file is routed to a VPS printer owned by Connect:Direct, VPS copies the spool file to a disk file and invoke the Connect:Direct API to transfer the file within the Connect:Direct-defined network.

- Copy to JES Print Queues

  Connect:Direct dynamically allocates a print file and writes the input file directly to the JES Spool.

- Copy to JES Reader Queues

  Connect:Direct dynamically allocates an internal reader and writes the input file directly to the JES Reader.

For outbound transfers, this feature requires the following additional products from Levi, Ray & Schoup (LRS):

- VTAM Printer Support (VPS)
- VPS Connect:Direct Interface (VPS/CDI) Option

Note: For sending output to the JES reader, you do not need the VTAM Printer Support and VPS Connect:Direct Interface (VPS/CDI) Option components. See *Chapter 15, Transferring Data to the JES Reader or Spool*, for more information.

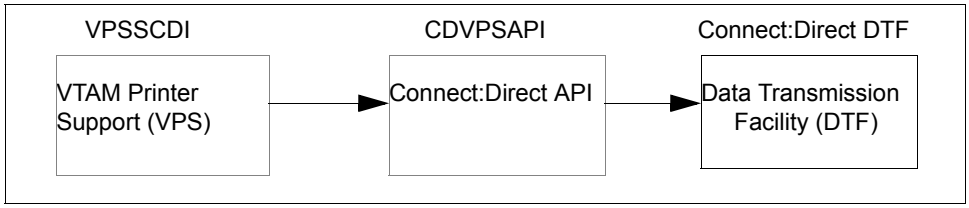
Connect:Direct Spool Transfer uses standard Connect:Direct facilities, which provide automation, reliability, management, interoperability, and security.

Spool Transfer Components

The major components of Connect:Direct Spool Transfer are:

- VTAM Printer Support (VPS) System
- Application Program Interface (API)
- Data Transmission Facility (DTF)

The following diagram shows the relationship between these components. This chapter summarizes the function of each component.



You implement Connect:Direct Spool Transfer by changing definitions in the VPS initialization. When a JES spool file is routed to a Connect:Direct virtual printer, VPS copies the spool file to disk and invokes Connect:Direct Spool Transfer to transfer the file within the Connect:Direct defined network.

Chapter 14

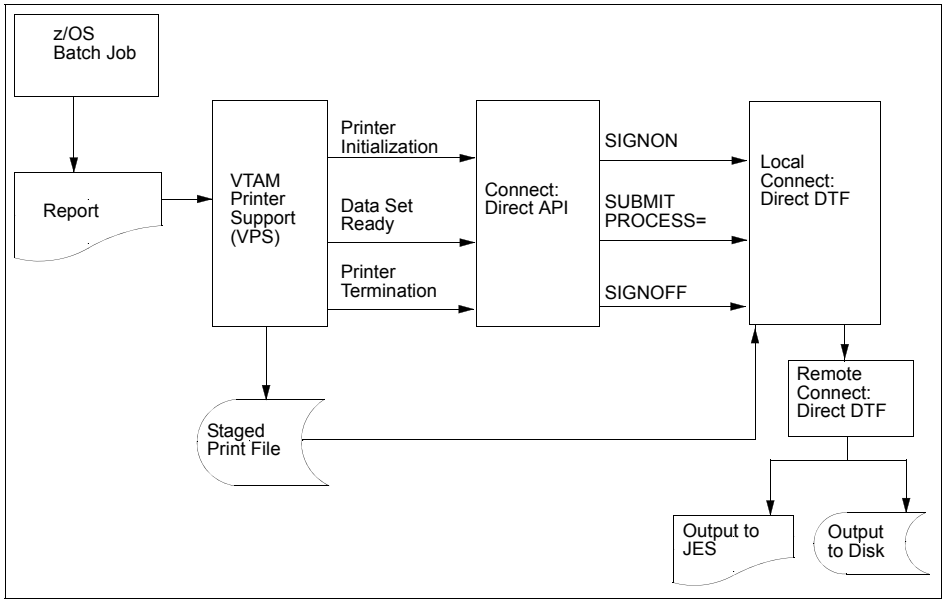
Receiving Spool Output

This chapter explains how VTAM Printer Support and the Connect:Direct API work together to distribute print files from JES Spool to Connect:Direct.

VTAM Printer Support

Connect:Direct for z/OS uses VTAM Printer Support (VPS) to retrieve JES spool files for processing. Expansion of the VPS application includes Connect:Direct support for moving print files from the JES spool to any Connect:Direct node. The VPS module (VPSSCDI) directs spooled output from JES to the Connect:Direct API for file transfer.

The following diagram illustrates the flow of the print file through this process.



System Control

VPS passes control to the Connect:Direct API at the following times:

- System initialization
- Printer initialization
- SYSOUT data set ready
- Printer termination
- System termination

You can change VPSSCDI to modify Connect:Direct control blocks prior to calling CDVPSAPI. Refer to *Customizing VPSSCDI* on page 120.

System Initialization

When a JES spool file is routed to a Connect:Direct-defined printer, VPS first transfers control to the Connect:Direct API for printer initialization. This action causes the API to issue a SIGNON to the local DTF.

Printer Initialization

Upon successful signon, VPS copies the print files to a disk file and passes this file to the Connect:Direct API.

SYSOUT Data Set Ready

The Connect:Direct API then constructs a SUBMIT PROCESS command to pass printer attributes, data set name, and the submitter's user ID to the local DTF.

If the SUBMIT fails because the submitted Process could not be located or the specified SNODE is unavailable, the print file is returned to the JES print queue according to the values specified in the VPS printer requeue parameters. If the SUBMIT fails for any other reason, VPS deletes the staged data set, drains the Connect:Direct printer, and leaves the JES spool file in the JES queue for restart.

If the SUBMIT is successful, VPS releases ownership of the staged data set to Connect:Direct, and the JES spool file is removed from JES.

Printer Termination

When no more JES files are printing to the Connect:Direct-defined printer, VPS calls the Connect:Direct API with a printer termination request.

System Termination

The Connect:Direct API then sends a SIGNOFF command to the local DTF.

Connect:Direct API

The Connect:Direct API (CDVPSAPI) communicates with the Data Transmission Facility through an application interface. CDVPSAPI signs on to the local DTF, submits a Process, and signs off of the local DTF.

CDVPSAPI

The Connect:Direct module CDVPSAPI, part of the VPS interface, is distributed with Connect:Direct in SAMPLIB. This program is in source form so that you can assemble and link-edit when you install a new release of Connect:Direct VPS.

This program utilizes the VPS system log for reporting exception conditions and informational messages describing the Processes submitted to the local DTF. This log serves as one central point to determine the status of any print file controlled by VPS.

The CDVPSAPI program receives control from VPSSCDI and validates the function calls and control block fields passed by VPSSCDI. If a function executes successfully, the program returns a condition code of **0** (zero) to the VPSSCDI.

If a noncritical error occurs, CDVPSAPI returns a code of **4** to VPSSCDI to indicate that VPS requeues the print file according to VPS printer specifications.

If CDVPSAPI detects a critical error, it returns a code of **8** or greater, indicating that VPS can EDRAIN the printer.

A function call of PRINTER INITIALIZATION causes a SIGNON to be sent to the local DTF as defined in the network map specification for this printer.

A function call of DATA SET SYSOUT READY invokes the routines to validate the control blocks and build the Connect:Direct SUBMIT PROCESS command and send it to the local DTF.

A function call of PRINTER TERMINATION sends a SIGNOFF to the local DTF.

CDVPSAPI uses the Connect:Direct API interface module (DMCHLAPI) to communicate with the Data Transmission Facility.

VPSSCDI

The VPS module VPSSCDI, the interface to Connect:Direct, is distributed as part of VPS. This program is in source form so that you can reassemble and link-edit it when you install a new release of VPS or Connect:Direct.

Connect:Direct SIGNON

The SIGNON command uses the network map specification for the specified Connect:Direct printer.

If you do not have a network map override specified in the VPS printer definition table (CDNETMAP), the SIGNON command uses the DMNETMAP DD statement in the VPS startup JCL for the network map.

SIGNON Security

The SIGNON command for CDVPSAPI does not pass a user id or password. If you are using the Stage 1 security exit, Connect:Direct extracts the user id associated with the VPS started task and puts the user id and a special password (assigned by the Stage 1 exit) into the UICB, a Connect:Direct user interface control block used for security.

If the Stage 1 processing is successful, the SIGNON command passes to the DTF, where the DTF Stage 2 security is invoked. The Stage 2 exit recognizes the special password, as assigned by the Stage 1 exit. All verification calls to the security system are by user id only.

If you are sending to a secured Connect:Direct, you can specify user id and password by adding the SNODEID specification to the Process. All Processes submitted by the VPS API have the security access level associated with the VPS job or started task, unless the security is overridden by PNODEID, SNODEID, and/or secure point-of-entry (SPOE) translation.

Process Names

VPS print files are routed through the Connect:Direct network by specifying the SNODE keyword on the submitted Process. To provide flexibility, you can set up the VPS Connect:Direct printer either to submit a single Process name regardless of printer class or to submit a Process name with the printer output class appended to the name.

By appending the printer class to the name, one VPS/CDI printer can submit up to 36 different Process names, one for each printer class A–Z or 0–9. You can control the Process name being submitted with the VPS printer name and VPS printer definitions as follows:

- Define the CDSNODE in the VPS Printer Definition Table
- Define the CDPMBR in the VPS Printer Definition Table
- Use a VPS printer name of seven characters or less
- Use a VPS printer name of eight characters

This flexibility enables a VPS printer to submit Processes that send print files to one or more Connect:Direct SNODEs.

Use the following table to select the method that is best for your environment.

Where to Code	Keyword	Requirements	Results
Use Default (Do not code)	None	1 to 7-character printer name	Process name is the VPS Printer name with the class appended.
Use Default (Do not code)	None	8-character printer name	Process name is the VPS Printer name.
VPS Printer Definition Table	CDSNODE	Value for CDSNODE Process name specified in CDPMBR	Process name is the printer name. (Class is not appended.)

Where to Code	Keyword	Requirements	Results
VPS Printer Definition Table	CDSNODE CDPMBR	Value for CDSNODE and CDPMBR	Process name is the name specified in CDPMBR. (Class is not appended.)
VPS Printer Definition Table	CDPMBR	8 characters	Process member name remains the same for all print files. (Class is not appended.)
VPS Printer Definition Table	CDPMBR	Up to 7 characters	Process name is the name specified in CDPMBR (Class is appended.)

Note: Connect:Direct searches the Connect:Direct Process library for the Process name. You must define the Process names in the Process library.

For example, if you select CDPMBR=CDPROC and the output print class is **A**, Connect:Direct searches the Process library for the Process name of CDPROCA. If you select an output print class of **C** for the same printer, Connect:Direct searches for the Process name CDPROCC.

However, if you use an eight-character name such as CDPMBR=CDPROC01, the Process name for that printer is CDPROC01, regardless of the printer class.

CDPROCES

The following sample Process is provided in the Install Process Library. This example shows all attributes that can pass from CDVPSAPI to the local Connect:Direct DTF in the SUBMIT statement. Any value of X'00' passed from VPS to CDVPSAPI is considered a null value, and the SUBMIT command does not pass the corresponding field to the Process. For example, if your printer output DD statement does not code COPIES, then the symbolic &COPIES is not passed in the SUBMIT command.

```
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*                                                                       */
/*      C:D-ZOS JES SPOOL TRANSFER FACILITY SAMPLE PROCESS               */
/*                                                                       */
/*   CHANGE $CD.PROC TO THE Z/OS PROC LIBRARY THAT YOU HAVE              */
/*   INSTALLED THE PROC GENER INTO.  THE CONNECT:DIRECT                  */
/*   INSTALLATION LOADED THIS MEMBER INTO YOUR                           */
/*   CONNECT:DIRECT PROCESS LIBRARY.                                     */
/*                                                                       */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
CDPROCES PROC   SNODE=&SNODE,                                    -
                &ADDR1=,         /* ADDRESS LINE 1         */ -
                &ADDR2=,         /* ADDRESS LINE 1         */ -
                &ADDR3=,         /* ADDRESS LINE 1         */ -
                &ADDR4=,         /* ADDRESS LINE 1         */ -
```

Continued

```
&BLDG=,          /* BUILDING               */ -
&BURST=,         /* BURST=YES OR NO        */ -
&CHARS=,         /* CHAR ARRANGEMENT TABLE */ -
&CKPTL=,         /* CKPTLINE               */ -
&CKPTP=,         /* CKPTPAGE               */ -
&CKPTS=,         /* CKPTSEC                */ -
&CLASS=,         /* OUTPUT CLASS           */ -
&CMOD=,          /* COPY MODIFICATION MOD  */ -
&CMTTC=,         /* COPY MODULE TABLE REF  */ -
&CNTL=,          /* DEFAULT SPACING        */ -
&COMPACT=,       /* COMPACTATION TABLE     */ -
&COPIES=,        /* OUTPUT NUMBER COPIES   */ -
&COPYG=,         /* COPY GROUP(S)          */ -
&DATCK=,         /* DATACK                 */ -
&DEFAULT=,       /* DEFAULT                */ -
&DEPT=,          /* DEPARTMENT             */ -
&DEST=,          /* OUTPUT DESTINATION     */ -
&FCB=,           /* OUTPUT WTR FCB         */ -
&FLASH=,         /* FLASH FORMS OVERLAY    */ -
&FLSCT=,         /* FLASH FORMS OVERLAY CNT*/ -
&FMDEF=,         /* FORMDEF                */ -
&FORM=,          /* OUTPUT FORM            */ -
&GROUPID=,       /* GROUPID                */ -
&INDEX=,         /* INDEX                  */ -
&JACCT=,         /* JOB ACCOUNTING NUMBER  */ -
&JESDS=,         /* JESDS                  */ -
&JOBID=,         /* JES ASSIGNED JOB ID    */ -
&JOBNM=,         /* JOB NAME               */ -
&JPNAME=,        /* JOB PROGRAMMERS NAME   */ -
&JPROC=,         /* JOB PROC NAME          */ -
&JROOM=,         /* JOB PROGRAMMERS ROOM   */ -
&JSECL=,         /* SECURITY LABEL         */ -
&LINDEX=,        /* LINDEX                 */ -
&LINECT=,        /* LINES PER PAGE         */ -
&NAME=,          /* NAME                   */ -
&NOTIFY1=,       /* 1ST NOTIFY ID          */ -
&NOTIFY2=,       /* 2ND NOTIFY ID          */ -
&NOTIFY3=,       /* 3RD NOTIFY ID          */ -
&NOTIFY4=,       /* 4TH NOTIFY ID          */ -
&OPTCD=,         /* OPTCD=J SPECIFIED      */ -
&PGDEF=,         /* PAGEDEF                */ -
&PIMCT=,         /* PIMSG MSG-COUNT        */ -
&PIMSG=,         /* PIMSG                  */ -
&PRMODE=,       /* PRMODE                 */ -
&PRTY=,          /* PRTY                   */ -
&ROOM=,          /* ROOM                   */ -
&STEPDD=,        /* STEP DDNAME            */ -
&STEPNM=,        /* STEP NAME              */ -
&SUBNAME=,       /* SUBMITTERS NAME        */ -
&THRES=,         /* THRESHLD               */ -
&TITLE=,         /* TITLE                  */ -
```

Continued

```
                &TRC=,           /* TRC                     */ -
                &UCS=,           /* OUTPUT WTR UCS          */ -
                &UDATA01=,       /*  1ST USERDATA           */ -
                &UDATA02=,       /*  2ND USERDATA           */ -
                &UDATA03=,       /*  3RD USERDATA           */ -
                &UDATA04=,       /*  4TH USERDATA           */ -
                &UDATA05=,       /*  5TH USERDATA           */ -
                &UDATA06=,       /*  6TH USERDATA           */ -
                &UDATA07=,       /*  7TH USERDATA           */ -
                &UDATA08=,       /*  8TH USERDATA           */ -
                &UDATA09=,       /*  9TH USERDATA           */ -
                &UDATA10=,       /* 10TH USERDATA           */ -
                &UDATA11=,       /* 11TH USERDATA           */ -
                &UDATA12=,       /* 12TH USERDATA           */ -
                &UDATA13=,       /* 13TH USERDATA           */ -
                &UDATA14=,       /* 14TH USERDATA           */ -
                &UDATA15=,       /* 15TH USERDATA           */ -
                &UDATA16=,       /* 16TH USERDATA           */ -
                &ULIB1=,         /*  1ST USERLIB LIBRARY    */ -
                &ULIB2=,         /*  2ND USERLIB LIBRARY    */ -
                &ULIB3=,         /*  3RD USERLIB LIBRARY    */ -
                &ULIB4=,         /*  4TH USERLIB LIBRARY    */ -
                &ULIB5=,         /*  5TH USERLIB LIBRARY    */ -
                &ULIB6=,         /*  6TH USERLIB LIBRARY    */ -
                &ULIB7=,         /*  7TH USERLIB LIBRARY    */ -
                &ULIB8=,         /*  8TH USERLIB LIBRARY    */ -
                &VPSDSN=,        /* VPS PRINTER STAGED DSN  */ -
                &WRITER=,        /* OUTPUT WRITER NAME      */ -
                &WTR=,           /* OUTPUT WTR NAME (OLD)   */ -
                &OUTDSN=&SUBNAME..&JOBNM..&JOBID..&STEPNM..&STEPDD
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*   COPY FILE STAGED BY VPSSCDI                                     */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
STEP01   COPY FROM(PNODE                                          -
               DSN=&VPSDSN                                        -
               DISP=SHR )                                         -
               CKPT=10M                                           -
               COMPRESS EXT                                       -
         TO(SNODE                                                 -
           DSN=&OUTDSN                                            -
               DISP=(NEW,CATLG,DELETE) )
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*   DELETE FILE STAGED BY VPSSCDI                                   */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
STEP02   IF     (STEP01 = 0) THEN
STEP03   RUN TASK (PGM=DMRTDYN                                    -
    PARM=(C"ALLOC DSN=&VPSDSN,DISP=(OLD,DELETE)"                  -
    F'-1'                                                         -
    C"UNALLOC DSN=&VPSDSN"))
         EIF
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*   EXECUTE PROC GENER TO PRINT THE OUTPUT FILE                     */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
STEP04   RUN TASK (PGM=DMRTSUB                                    -
    PARM=("DSN=$CD.PROC(GENER),DISP=SHR",                         -
          "ADDR1     &ADDR1",                                     -
          "ADDR2     &ADDR2",                                     -
          "ADDR3     &ADDR3",                                     -
          "ADDR4     &ADDR4",                                     -
          "BLDG      &BLDG",                                      -
```

Continued

```
            "BURST     &BURST",                                  -
            "CHARS     &CHARS",                                  -
            "CKPTL     &CKPTL",                                  -
            "CKPTP     &CKPTP",                                  -
            "CKPTS     &CKPTS",                                  -
            "CLASS     &CLASS",                                  -
            "CMOD      &CMOD",                                   -
            "CMTTC     &CMTTC",                                  -
            "CNTL      &CNTL",                                   -
            "COMPACT   &COMPACT",                                -
            "COPIES    &COPIES",                                 -
            "COPYG     &COPYG",                                  -
            "DATCK     &DATCK",                                  -
            "DEFAULT   &DEFAULT",                                -
            "DEPT      &DEPT",                                   -
            "DEST      &DEST",                                   -
            "FCB       &FCB",                                    -
            "FLASH     &FLASH",                                  -
            "FLSCT     &FLSCT",                                  -
            "FMDEF     &FMDEF",                                  -
            "FORM      &FORM",                                   -
            "GROUPID   &GROUPID",                                -
            "INDEX     &INDEX",                                  -
            "INPUT     &OUTDSN",                                 -
            "JACCT     &JACCT",                                  -
            "JESDS     &JESDS",                                  -
            "JOBID     &JOBID",                                  -
            "JOBNM     &JOBNM",                                  -
            "JPNAME    &JPNAME",                                 -
            "JPROC     &JPROC",                                  -
            "JROOM     &JROOM",                                  -
            "JSECL     &JSECL",                                  -
            "LINDEX    &LINDEX",                                 -
            "LINECT    &LINECT",                                 -
            "NAME      &NAME",                                   -
            "NOTIFY1   &NOTIFY1",                                -
            "NOTIFY2   &NOTIFY2",                                -
            "NOTIFY3   &NOTIFY3",                                -
            "NOTIFY4   &NOTIFY4",                                -
            "OPTCD     &OPTCD",                                  -
            "PGDEF     &PGDEF",                                  -
            "PIMCT     &PIMCT",                                  -
            "PIMSG     &PIMSG",                                  -
            "PRMODE    &PRMODE",                                 -
            "PRTY      &PRTY",                                   -
            "ROOM      &ROOM",                                   -
            "STEPDD    &STEPDD",                                 -
            "STEPNM    &STEPNM",                                 -
            "SUBNAME   &SUBNAME",                                -
            "THRES     &THRES",                                  -
            "TITLE     &TITLE",                                  -
            "TRC       &TRC",                                    -
            "UCS       &UCS",                                    -
            "UDATA01   &UDATA01",                                -
            "UDATA02   &UDATA02",                                -
            "UDATA03   &UDATA03",                                -
```

Continued

```
            "UDATA04   &UDATA04",                                -
            "UDATA05   &UDATA05",                                -
            "UDATA06   &UDATA06",                                -
            "UDATA07   &UDATA07",                                -
            "UDATA08   &UDATA08",                                -
            "UDATA09   &UDATA09",                                -
            "UDATA10   &UDATA10",                                -
            "UDATA11   &UDATA11",                                -
            "UDATA12   &UDATA12",                                -
            "UDATA13   &UDATA13",                                -
            "UDATA14   &UDATA14",                                -
            "UDATA15   &UDATA15",                                -
            "UDATA16   &UDATA16",                                -
            "ULIB1     &ULIB1",                                  -
            "ULIB2     &ULIB2",                                  -
            "ULIB3     &ULIB3",                                  -
            "ULIB4     &ULIB4",                                  -
            "ULIB5     &ULIB5",                                  -
            "ULIB6     &ULIB6",                                  -
            "ULIB7     &ULIB7",                                  -
            "ULIB8     &ULIB8",                                  -
            "VPSDSN    &VPSDSN",                                 -
            "WRITER    &WRITER",                                 -
            "WTR       &WTR",                                    -
          ))  SNODE
          EXIT
```

GENER

The following sample JCL is provided in the Install Process Library. This example is the GENER job that is submitted by DMRTSUB in STEP 04 of the preceding Process example. Because it runs on the SNODE, it shows how you can print files back into the JES spool at the remote site.

If Connect:Direct Spool Transfer is enabled at the remote site, you can use Connect:Direct to transfer output back into the JES spool. Refer to *Sending Output to the JES Reader* on page 124.

```
//&JOBNM     JOB (00000),&PGMR,PRTY=12,TIME=(10),
//REGION=4096K,MSGLEVEL=(1,1),MSGCLASS=X
//*
//STEP01   EXEC PGM=IEBGENER
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD DUMMY
//SYSUT1    DD DSN=&INPUT,
//          DISP=(OLD,DELETE,KEEP)
//SYSUT2    DD SYSOUT=&CLASS,COPIES=&COPIES,DEST=&DEST
//
```

Caution: When you use this JCL to test your own VPS, a print loop can occur if you use the symbolic &DEST on the same VPS system from which you submitted the print job.

Symbolic Definitions

This section identifies the parameters that can be passed to the Process submitted by the Connect:Direct API.

Job and Jobstep Values

The following table lists the job and jobstep values.

Parameter	Definition
&JACCT	Job accounting number
&JOBID	JOB ID assigned by JES
&JOBNM	JOB name
&JPNAME	Name of the programmer
&JPROC	Procedure name
&JROOM	Room number of the programmer
&JSECL	Security label
&STEPDD	DD name of the print file DD statement
&STEPNM	Process step name
&SUBNAME	User ID of the submitter
&VPSDSN	Printer staged data set

Printer File Attributes

Refer to *SYSOUT Keyword* on page 129 for the available parameters and associated symbolic names for print file attributes passed by the VPSSCDI interface program. Any non-zero attributes for a printer file cause the corresponding symbolic name to be generated on the SUBMIT command created by the Connect:Direct interface program, CDVPSAPI. You can reference these symbolic names in the Connect:Direct Process you are submitting.

Customizing VPSSCDI

VPSSCDI creates the control blocks that the Connect:Direct API uses to build the SIGNON, SUBMIT, and SIGNOFF commands. Any customized validation or modification to the Connect:Direct control blocks is done at this point.

You can change printer values to reflect the destination environment. For example, you can change the destination name to a printer name at your location.

You can perform security or usage validation by validating any fields and either continuing the processing or returning to VPS with the appropriate return code settings.

The VPS return code settings are:

- RC=00 Normal execution continues.
- RC=04 Print file is requeued per VPS printer specifications.
- RC=08 The printer is EDRAINed.

Note: Do not change VPS control blocks.

Chapter 15

Transferring Data to the JES Reader or Spool

Connect:Direct Spool Transfer enables you to transfer data from any Connect:Direct node to the JES reader or the JES spool. You can route job control language (JCL) statements to the JES reader or output files to the JES spool.

This chapter explains how to use Connect:Direct Spool Transfer to transfer data to the JES reader.

Using the JES Reader or Spool

By transferring data to the JES reader, any Connect:Direct node can send z/OS JCL to the JES reader. Unlike the Connect:Direct RUN JOB, this feature enables z/OS JCL to reside at the location executing the Process on any media that Connect:Direct supports.

The RUN JOB Security Exit, which provides a standard interface for security verification of job streams before they are submitted, is fully supported. For additional information on the RUN JOB Security Exit, see the *Implementing Security* chapter in the *Connect:Direct for z/OS Administration Guide*.

Dynamic Allocation

Connect:Direct Spool Transfer uses dynamic allocation (SVC 99) to allocate the JES spool file. You can also use the OUTADD and OUTDEL macros to allocate an OUTPUT DD reference statement dynamically.

Checkpoint/Restart

Connect:Direct Spool Transfer supports the checkpoint/restart facility of Connect:Direct. Connect:Direct takes a checkpoint at the *top of form* of the page printed after the checkpoint interval. *Top of form* is one of the following:

- X'F1' for ASA carriage control
- X'8B' for machine carriage control
- LINECT (line count) if NOCC or the TOF=X'xx' is specified in the Process

Banner Page

You can print a Connect:Direct banner page at the beginning of each print report. For information on how to use this feature, refer to *Connect:Direct Banners* on page 125.

Sending Output to the JES Reader

Use Connect:Direct syntax to transfer data to the JES reader. To initiate this transfer, specify the keyword, READER, in the COPY TO Process statement for Connect:Direct nodes that have enabled Spool transfer. Keyword and parameter definitions are in the section, *Connect:Direct Syntax* on page 128.

Some older releases of Connect:Direct do not support the READER keyword. Use the SYSOPTS parameter OUTPUT=READER for such nodes. Keyword and parameter definitions are in *Connect:Direct Syntax* on page 128.

Examples

You can adapt the following Process language examples to output JCL into the JES reader. These examples are in the Sample Library.

Using the READER Keyword

The following sample Process (CDTORDR2) uses the READER keyword to output JCL into the JES reader.

```
CDTORDR2 PROCESS   SNODE=xx.xxx.xxxxx                          -
STEP01   COPY FROM(PNODE                                       -
                   DSN=SAMPLE.JCL.LIB(JCL)                     -
                   DISP=SHR                                    -
                  )                                            -
                TO(SNODE                                       -
                   READER                                      -
                  )
         EXIT
```

Using the SYSOPTS

The following sample Process (CDTORDR1) uses SYSOPTS to output JCL into the JES reader.

```
CDTORDR1  PROCESS   SNODE=xx.xxx.xxxxx                        -
STEP01    COPY FROM(PNODE                                     -
                    DSN=SAMPLE.JCL.LIB(JCL)                   -
                    DISP=SHR                                  -
                   )                                          -
                  TO(SNODE                                    -
                    DSN=NULLFILE                              -
                    DISP=RPL                                  -
                    SYSOPTS="OUTPUT=READER"                   -
                   )
          EXIT
```

Sending Output to JES Spool Files

You can also use Connect:Direct syntax to direct files transferred through Connect:Direct directly to the JES2/JES3 spool.

Copying a file into the JES spool is controlled through the SYSOUT=(....,....) keyword on the COPY TO Process statement. Keyword and parameter definitions are in *Connect:Direct Syntax* on page 128.

You can specify all keywords available on the z/OS JCL SYSOUT DD statement within the Connect:Direct SYSOUT=(...) Process keyword. The format of all subparameters within the Connect:Direct SYSOUT=(...) follows the rules in the *OS/390 JCL Reference Guide* for that keyword.

Note: You cannot use SYSOUT=class because the Connect:Direct feature *requires* that you specify all subparameters within parentheses. Refer to page 127 for examples using this keyword.

Connect:Direct Banners

You can print a Connect:Direct banner page at the beginning of each print report. If you selected the checkpoint/restart option for the file transfer, the word RESTARTED is added to the banner page during restart.

Select this option by coding the BANNER=(...) subparameter of the SYSOUT=(...) Process keyword. Banner values are supplied in pairs of literal = value. One banner line is printed for each supplied pair. Keyword and parameter definitions are in *Connect:Direct COPY Process* on page 127.

If the value contains a character other than 0–9 or A–Z, you must enclose it in quotation marks. For example, PROGRAMMER=John Doe must be specified as PROGRAMMER="John Doe". JOBNAME=OS/400 must be specified JOBNAME="OS/400".

You can use literals of 1–25 characters. Literals longer than 25 characters are truncated. Values can be 1–30 characters. Values longer than 30 characters are truncated. If the literal or value contains a character other than 0–9 or A–Z, you must enclose it in quotation marks.

In the following example, three banner data lines are printed. JUNK is ignored because it is not paired with a value.

```
BANNER=(PROGRAMMER=&JPNAME,JOBNAME=&JOBNM,                          -
                   SUBMITTER=&SUBNAME,STEPNAME=&STEPNM,             -
                   DDNAME=&STEPDD,JUNK)
```

Assuming that the symbolics are set to the values defined previously, the following example is generated.

```
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****   PROGRAMMER                JOHN DOE                       ****
****   JOBNAME                   JOHN1X                         ****
****   SUBMITTER                 JOHN1                          ****
****   STEPNAME                  STEP1                          ****
****   DDNAME                    SYSUT2                         ****
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
```

Banner After Restart

The following example illustrates the banner that prints if the job is restarted.

```
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
******************      R E S T A R T E D      ********************
****   PROGRAMMER                JOHN DOE                       ****
****   JOBNAME                   JOHN1X                         ****
****   SUBMITTER                 JOHN1                          ****
****   STEPNAME                  STEP1                          ****
****   DDNAME                    SYSUT2                         ****
******************      R E S T A R T E D      ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
****************** C O N N E C T : D I R E C T ********************
```

Connect:Direct COPY Process

The Connect:Direct COPY Process controls the printing of the output data set by specifying key values within the new Process keyword SYSOUT=(...) or by using the SYSOPTS=(SYSOUT=(...)) for platforms other than z/OS.

Examples

The following sample Process shows the simplest form of a COPY Process. SYSOUT=A is not a valid value in this example. You must put all subparameters in parentheses, and you must set CLASS to the output class assigned to the print file.

```
TESTP    PROCESS  SNODE=
STEP01   COPY     FROM(DSN=NAME,DISP=SHR)                        -
                    TO(SYSOUT=(CLASS=A)                          -
                       )
         END
```

The next example is an alternative way to code the same Process. In this example, the CLASS= parameter is not required because of the additional parentheses in the first parenthetical group of the SYSOUT keyword. This example corresponds to the syntax of the SYSOUT DD statement.

```
TESTP    PROCESS  SNODE=
STEP01   COPY     FROM(DSN=NAME,DISP=SHR)                        -
                    TO(SYSOUT=((A,writer-name,form-name),...)    -
                       )
         END
```

The following example uses the SYSOUT= keyword and its subparameters within the SYSOPTS keyword.

```
TESTP   PROCESS  SNODE=
STEP01  COPY     FROM(DSN=NAME,DISP=SHR)                         -
                   TO(DSN=NULLFILE                               -
                      SYSOPTS="(SYSOUT=(CLASS=A))"               -
                      )
        END
```

Carriage Control

The input file DCB RECFM specification determines carriage control. If RECFM is VA, VBA, FA, FBA, or UA, ASA carriage control is assigned (ASA is also the default). If RECFM is VM, VBM, FM, FBM, or UM, then machine carriage control is assigned.

You can optionally override the specification by defining the NOCC, LINECT, and TOF parameters in the SYSOUT keyword CC=. Keyword and parameter definitions are in the following section, *Connect:Direct Syntax* on page 128.

The following example shows how to override carriage control specifications.

```
TESTP    PROCESS  SNODE=
STEP01   COPY     FROM(DSN=NAME,DISP=SHR)                          -
                    TO(SYSOUT=(CLASS=A,                            -
                       CC=(NOCC,LINECT=n,TOF=x'F1'))                   -
                       )
         END
```

Connect:Direct Syntax

All keywords and parameters of the COPY TO statement that support Spool Transfer are defined in this section. The table provided in the Symbolics Definitions section identifies where the parameters of the SYSOUT=(...) keyword are documented. For detailed information on other keywords and parameters, see the Connect:Direct Processes Web site at http://www.sterlingcommerce.com/Documentation/processes/processhome.html.

Keyword	Description
READER	Specifies the output data set that is directed to the JES reader.
SYSOUT=(...)	Controls the printing of the output data set.
BANNER= ('literal 1' = value1['literal 2' = value2[,...]])	Specifies the banner values in pairs of literal = value. One banner line is printed for each supplied pair. A literal without a corresponding value is ignored. You can use literals of 1–25 characters. Literals greater than 25 characters are truncated. You can use a value size of 1–30 characters. Values greater than the 30 characters are truncated. If the name or value contains a character other than 0–9 or A–Z, you must enclose it in quotation marks.
CC=(A|M|NOCC|LINECT=|TOF=)	Overrides carriage control supplied by the input file DCB RECFM. **A** (for ASA) is assigned if RECFM is VA, VBA, FA, FBA, or UA. It is also the default if carriage control is not specified. **M** (for machine) is assigned if RECFM is VM, VBM, FM, FBM, or UM. **NOCC** identifies the file as having no carriage control. If you specify NOCC, you must also specify LINECT and TOF. **LINECT=nnn** identifies the number of lines printed per page. The default is **55**. **TOF=X'F1'** identifies the hex character that is inserted in the print line at the interval specified in the LINECT parameter. The hex character specified by the TOF parameter indicates to skip to the top of form. The default is **X'F1**'.
SYSOPTS="..."	**OUTPUT=READER** specifies that the output data set is directed to the JES reader. **SYSOUT=(class=class)|(class,writer-name,form-name)** specifies that the output data set is directed to the JES spool files on platforms other than z/OS.

Symbolic Definitions

This section lists the parameters that you can pass to JES from the Connect:Direct API.

SYSOUT Keyword

The following table lists the subparameter fields for the SYSOUT keyword. The table identifies the format for the parameter, the VPS symbolic name, and identifies where the parameter is documented.

The *As Documented by* column states where you can find additional information about the parameters. *Spool Transfer* indicates that the parameter is documented in this book. References to JCL parameters indicate that additional information is available in the JCL documentation.

Parameter	Format	VPS Symbolic Name	As Documented by
ADDRESS=	(‘address line1’,address line2’,address line3, address line4)	&ADDR1,&ADDR2, &ADDR3,&ADDR4	SYSOUT DD
BANNER=	(‘literal 1’ = value1, ‘literal 2’ = value2,...)		Spool Transfer
BUILDING=	‘building identification’	&BLDG	OUTPUT DD
BURST=	Yes or No	&BURST	OUTPUT DD & SYSOUT DD
CC=	ASA or Machine or NOCC		Spool Transfer
CHARS=	(tablename,tablename...)	&CHARS	OUTPUT DD & SYSOUT DD
CKPTLINE=	nnnnn	&CKPTL	OUTPUT DD
CKPTPAGE=	nnnnn	&CKPTP	OUTPUT DD
CKPTSEC=	nnnnn	&CKPTS	OUTPUT DD
CLASS=	class	&CLASS	OUTPUT DD
COMPACT=	compaction-table-name	&COMPACT	OUTPUT DD
COMSETUP=	name		OUTPUT DD
CONTROL=	PROGRAM SINGLE DOUBLE TRIPLE	&CNTL	OUTPUT DD
COPIES=	nnn (,(group value,group value,...))	&COPIES &COPYG	SYSOUT DD OUTPUT DD

Parameter	Format	VPS Symbolic Name	As Documented by
DATACK=	BLOCK UNBLOCK BLKCHAR BLKPDS	&DATCK	OUTPUT DD
DEFAULT=	Yes or No	&DEFAULT	OUTPUT DD
DEPT=	'department identification'	&DEPT	OUTPUT DD
DEST=	destination (nodename,userid)	&DEST	SYSOUT DD & OUTPUT DD
DPAGELBL=	Yes or No		OUTPUT DD
FCB=	fcb-name (fcb-name,align | verify)	&FCB	SYSOUT DD & OUTPUT DD
FLASH=	(overlay-name,count) (,count)	&FLASH &FLSCT	SYSOUT DD & OUTPUT DD
FORMDEF=	membername	&FMDEF	OUTPUT DD
FORMS=	form-name	&FORM	OUTPUT DD
GROUPID=	output-group	&GROUPID	OUTPUT DD
HOLD=	Yes or No		SYSOUT DD
INDEX=	nn	&INDEX	OUTPUT DD
JESDS=	ALL JCL LOG MSG	&JESDS	OUTPUT DD
JOBNAME=	job name	&JOBNM	Spool Transfer
LINDEX	nn	&LINDEX	OUTPUT DD
LINECT=	nnn	&LINECT	OUTPUT DD
MODIFY=	module-name (module-name,trc)	&CMOD &CMTTC	SYSOUT DD & OUTPUT DD
NAME=	name of output separator	&NAME	OUTPUT DD
NOTIFY=	node.userid (node.userid,node.userid,...)	&NOTIFY1–4	OUTPUT DD
OPTJ=	Yes or No	&OPTCD	Spool Transfer
OUTBIN=	nnnnn		OUTPUT DD
OUTDISP=	(normal,abnormal)		OUTPUT DD
OUTPUT=	(reference)		SYSOUT DD

Parameter	Format	VPS Symbolic Name	As Documented by
PAGEDEF=	membername	&PGDEF	OUTPUT DD
PIMSG=	(yes,msg-count) (no,msg-count)	&PIMSG &PIMCT	OUTPUT DD
PRMODE=	line page process-mode	&PRMODE	OUTPUT DD
PRTY=	nnn	&PRTY	OUTPUT DD
ROOM=	‘room identification’	&ROOM	OUTPUT DD
SYSAREA=	Yes No		OUTPUT DD
SYSOUT=	(class) (class,writer-name, form-name)	&CLASS &WRITER &FORM	SYSOUT DD
THRESHLD=	nnnnnnnn	&THRES	OUTPUT DD
TITLE=	‘description of output’	&TITLE	OUTPUT DD
TOF=	X’F1’		Spool Transfer
TRC=	Yes No	&TRC	OUTPUT DD
UCS=	character-set-code (character-set-code,fold, verify)	&UCS	OUTPUT DD SYSOUT DD
USERDATA=	(‘user data description’, ‘user data description’,...)	&UDATA01–16	OUTPUT DD
USERLIB=	(‘library name 1’, ‘library name 2’,...)	&ULIB1–8	OUTPUT DD
WRITER=	name	&WRITER	OUTPUT DD

Sample Process (CDTOJES)

The following sample Process shows all attributes that you can specify within the SYSOUT=(...) keyword. This sample Process is provided in the Sample Library.

```
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*                                                                     */
/*             * * *  S A M P L E   P R O C E S S  * * *                */
/*                                                                     */
/*       C:D-ZOS JES SPOOL TRANSFER FACILITY SAMPLE PROCESS      */
/*                 INPUT FROM JES SPOOL TRANSFER FACILITY              */
/*                 OUTPUT TO VPS CONTROLLED PRINTER                    */
/*                                                                     */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
CDTOJES  PROC   SNODE=&SNODE,                                    -
                &ADDR1=,         /* ADDRESS LINE 1         */ -
                &ADDR2=,         /* ADDRESS LINE 1         */ -
                &ADDR3=,         /* ADDRESS LINE 1         */ -
                &ADDR4=,         /* ADDRESS LINE 1         */ -
                &BLDG=,          /* BUILDING               */ -
                &BURST=,         /* BURST=YES OR NO        */ -
                &CHARS=,         /* CHAR ARRANGEMENT TABLE */ -
                &CKPTL=,         /* CKPTLINE               */ -
                &CKPTP=,         /* CKPTPAGE               */ -
                &CKPTS=,         /* CKPTSEC                */ -
                &CLASS=,         /* OUTPUT CLASS           */ -
                &CMOD=,          /* COPY MODIFICATION MOD  */ -
                &CMTTC=,         /* COPY MODULE TABLE REF  */ -
                &CNTL=,          /* DEFAULT SPACING        */ -
                &COMPACT=,       /* COMPACTATION TABLE     */ -
                &COPIES=,        /* OUTPUT NUMBER COPIES   */ -
                &COPYG=,         /* COPY GROUP(S)          */ -
                &DATCK=,         /* DATACK                 */ -
                &DEFAULT=,       /* DEFAULT                */ -
                &DEPT=,          /* DEPARTMENT             */ -
                &DEST=,          /* OUTPUT DESTINATION     */ -
                &FCB=,           /* OUTPUT WTR FCB         */ -
                &FLASH=,         /* FLASH FORMS OVERLAY    */ -
                &FLSCT=,         /* FLASH FORMS OVERLAY CNT*/ -
                &FMDEF=,         /* FORMDEF                */ -
                &FORM=,          /* OUTPUT FORM            */ -
                &GROUPID=,       /* GROUPID                */ -
                &INDEX=,         /* INDEX                  */ -
                &JACCT=,         /* JOB ACCOUNTING NUMBER  */ -
                &JESDS=,         /* JESDS                  */ -
                &JOBID=,         /* JES ASSIGNED JOB ID    */ -
                &JOBNM=,         /* JOB NAME               */ -
                &JPNAME=,        /* JOB PROGRAMMERS NAME   */ -
                &JPROC=,         /* JOB PROC NAME          */ -
                &JROOM=,         /* JOB PROGRAMMERS ROOM   */ -
                &JSECL=,         /* SECURITY LABEL         */ -
```

Continued

```
         &LINDEX=,         /* LINDEX                 */ -
         &LINECT=,         /* LINES PER PAGE         */ -
         &NAME=,           /* NAME                   */ -
         &NOTIFY1=,        /* 1ST NOTIFY ID          */ -
         &NOTIFY2=,        /* 2ND NOTIFY ID          */ -
         &NOTIFY3=,        /* 3RD NOTIFY ID          */ -
         &NOTIFY4=,        /* 4TH NOTIFY ID          */ -
         &OPTCD=,          /* OPTCD=J SPECIFIED      */ -
         &PGDEF=,          /* PAGEDEF                */ -
         &PIMCT=,          /* PIMSG MSG-COUNT        */ -
         &PIMSG=,          /* PIMSG                  */ -
         &PRMODE=,         /* PRMODE                 */ -
         &PRTY=,           /* PRTY                   */ -
         &ROOM=,           /* ROOM                   */ -
         &STEPDD=,         /* STEP DDNAME            */ -
         &STEPNM=,         /* STEP NAME              */ -
         &SUBNAME=,        /* SUBMITTERS NAME        */ -
         &THRES=,          /* THRESHLD               */ -
         &TITLE=,          /* TITLE                  */ -
         &TOF=X'F1',       /* ASA TOP OF FORM        */ -
         &TRC=,            /* TRC                    */ -
         &UCS=,            /* OUTPUT WTR UCS         */ -
         &UDATA01=,        /*  1ST USERDATA          */ -
         &UDATA02=,        /*  2ND USERDATA          */ -
         &UDATA03=,        /*  3RD USERDATA          */ -
         &UDATA04=,        /*  4TH USERDATA          */ -
         &UDATA05=,        /*  5TH USERDATA          */ -
         &UDATA06=,        /*  6TH USERDATA          */ -
         &UDATA07=,        /*  7TH USERDATA          */ -
         &UDATA08=,        /*  8TH USERDATA          */ -
         &UDATA09=,        /*  9TH USERDATA          */ -
         &UDATA10=,        /* 10TH USERDATA          */ -
         &UDATA11=,        /* 11TH USERDATA          */ -
         &UDATA12=,        /* 12TH USERDATA          */ -
         &UDATA13=,        /* 13TH USERDATA          */ -
         &UDATA14=,        /* 14TH USERDATA          */ -
         &UDATA15=,        /* 15TH USERDATA          */ -
         &UDATA16=,        /* 16TH USERDATA          */ -
         &ULIB1=,          /*  1ST USERLIB LIBRARY   */ -
         &ULIB2=,          /*  2ND USERLIB LIBRARY   */ -
         &ULIB3=,          /*  3RD USERLIB LIBRARY   */ -
         &ULIB4=,          /*  4TH USERLIB LIBRARY   */ -
         &ULIB5=,          /*  5TH USERLIB LIBRARY   */ -
         &ULIB6=,          /*  6TH USERLIB LIBRARY   */ -
         &ULIB7=,          /*  7TH USERLIB LIBRARY   */ -
         &ULIB8=,          /*  8TH USERLIB LIBRARY   */ -
         &VPSDSN=,         /* VPS PRINTER STAGED DSN */ -
         &WRITER=,         /* OUTPUT WRITER NAME     */ -
         &WTR=,            /* OUTPUT WTR NAME (OLD)  */ -
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  */
```

Continued

```
/*   COPY FILE FROM VPS STAGED DATASET INTO JES SPOOL               */
/*        DESTINATION HAS BEEN CHANGED TO A VPS CONTROLLED PRINTER. */
/*        BANNER PAGE WILL BE PRODUCED.                             */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  */
STEP01   COPY FROM(                                    PNODE  -
               DSN=&VPSDSN                                        -
               DISP=SHR )                                         -
               CKPT=1M                                            -
               COMPRESS EXT                                       -
             TO(                                       SNODE  -
               SYSOUT=(CLASS=&CLASS,COPIES=&COPIES,DEST=&DEST -
               BANNER=(PROGRAMMER=&JPNAME,JOBNAME=&JOBNM,    -
               SUBMITTER=&SUBNAME,STEPNAME=&STEPNM,          -
               DDNAME=&STEPDD),JOBNAME=&JOBNM)               -
               )

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  */
/*   DELETE VPS STAGE INPUT FILE                                   */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  */
STEP02   IF     (STEP01 = 0) THEN
STEP03   RUN TASK (PGM=DMRTDYN                                  -
    PARM=(C"ALLOC DSN=&VPSDSN,DISP=(OLD,DELETE)"                -
    F'-1'                                                       -
    C"UNALLOC DSN=&VPSDSN"))
         EIF
         EXIT‘
```

Index

Symbols

A

B

C

D

E

F

G

H

I

J

M

N

O

P

Q

R

S

T

V

W

Z