Connect:Direct®
for VSE/ESA

User's Guide

Version 3.2

Sterling Commerce
An IBM Company

Connect:Direct for VSE/ESA User's Guide
Version 3.2

Second Edition

Sterling Commerce, Inc.

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

Contents

Chapter 4 Using Environment Commands 53

Chapter 5 Managing Processes 71

Chapter 6 Using Utility Programs 89

Chapter 7 Using VSE/POWER Support 101

Chapter 8 Using VSE Librarian Support 109

Glossary 113

Index 119

Chapter 1

About Connect:Direct for VSE/ESA

Connect:Direct links technologies and moves all types of information between networked systems/computers. It manages high-performance transfers by providing such features as: automation, reliability, efficient use of resources, application integration, and 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.

Connect:Direct allows data centers within and across networks to:

- Move large amounts of data
- Share information
- Schedule related application activities
- Automate data distribution
- Control and audit network activities
- Maintain network security
- Use a common command structure and environment-specific interfaces

Connect:Direct goes beyond traditional file transfer systems by eliminating the time-consuming, error-prone operator procedures associated with moving data. Connect:Direct capabilities extend from basic data movement functions to the management of data movement activities.

Connect:Direct for VSE/ESA Commands

You can issue Connect:Direct for VSE/ESA commands through:

- Batch Interface

  The batch interface enables you to issue Connect:Direct for VSE/ESA commands through the use of the DMBATCH program in a batch environment.

- Operator Interface

  The operator interface allows you to issue Connect:Direct for VSE/ESA commands from the VSE console. The *Connect:Direct Console Operator's Guide* describes the Connect:Direct for VSE/ESA operator interface.

✦ CICS Interactive User Interface

The interactive user interface (IUI) is a screen and dialog component that runs under the Customer Information Control System (CICS), using the CICS interface. For more information, refer to the *Connect:Direct for VSE/ESA User's Guide for CICS*.

✦ User-Written Applications

The high-level interface (DMCHLAPI) allows you to design an application that can communicate with Connect:Direct for VSE/ESA while running in a separate partition.

This manual contains descriptions of each of the Connect:Direct for VSE/ESA commands. Command function, format, parameters, and use through the batch interface provide the basis for the command descriptions. Examples demonstrate the different activities that you can perform using these commands.

The following table lists the Connect:Direct for VSE/ESA commands in alphabetical order.

Note: If you are running Connect:Direct for VSE/ESA on a platform that uses lowercase commands, you must set the case sensitivity option with the CASE initialization parameter. For example:

```
CASE=YES
```

Command	**Function**
CHANGE PROCESS	Modify a Process in the Transmission Control Queue (TCQ). See Chapter 5, *Managing Processes* for information about the TCQ.
DELETE PROCESS	Remove a Process from the Transmission Control Queue (TCQ). See Chapter 5, *Managing Processes* for information about the TCQ.
FLUSH PROCESS	Terminate an executing Process
MODIFY	Run Connect:Direct diagnostics
SELECT NETMAP	Choose for display the defined Connect:Direct nodes from the Network Map file
SELECT PROCESS	Examine a Process in the Transmission Control Queue (TCQ). See Chapter 5, *Managing Processes* for information about the TCQ.
SELECT STATISTICS	Retrieve and examine records in the Statistic Log
SELECT TYPE	Examine the records in the Type Defaults file
SIGNOFF	Terminate a connection to the Connect:Direct DTF
SIGNON	Connects to the Connect:Direct DTF
STOP CD	Stop Connect:Direct operation
SUBMIT	Submit a Process for execution
SUSPEND PROCESS	Terminate Process execution
SWAP NODE	Swap to another Connect:Direct node

Chapter 2

Signing On and Off

This chapter details signing on and off Connect:Direct for VSE/ESA. The following commands are explained:

- SIGNON
- Multiple Session SIGNON
- SWAP NODE
- SIGNOFF

Signing On to Connect:Direct for VSE/ESA

The SIGNON command connects you to Connect:Direct so that you can issue Connect:Direct commands. You can issue a SIGNON command by one of the following methods:

- Batch Interface
- User-Written Applications
- Operator Interface

Refer to the *Connect:Direct Console Operator's Guide* for information on using the Operator Interface.

Command Format

Parameters for the SIGNON command follow:

Label	Parameters	Command
(optional)	**SIGNON**	**NETMAP=network map file name**
		USERID=(userid, pswd[,newpswd])
		CASE=YES | NO
		ESF=YES | NO
		FOLD=YES | NO
		NODE=nodename
		PACCT='pnode accounting data'
		SACCT='snode accounting data'
		TMPDD=temporary data set dlblname
		TMPDSN=temporary data set name

Required Parameters

The following parameters are required for the SIGNON command.

Parameter	Description
NETMAP = network map file name	specifies the Connect:Direct network map file name, which is a 1-44-character alphanumeric string with the first character alphabetic. Ask your system programmer for the appropriate file name for the network map.
USERID = (userid, pswd[,newpswd])	specifies security information about the user who is signing on. **userid** specifies the 1-8-character security ID that will be passed to a security exit. **pswd** specifies the 1-8-character alphanumeric current security password. The security exit uses this parameter to validate the current security password. **newpswd** specifies the new security password. The security exit uses this optional parameter (1-8-character alphanumeric) to change the current security password to the new security password. It can contain 1-8 alphanumeric characters.

Optional Parameters

The following parameters used with the SIGNON command are optional:

Parameter	Description
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. NO is the default. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
ESF = YES | NO	specifies whether the Connect:Direct Extended Submit Facility (ESF) will be available to the user who is signing on. The ESF, which is an optional Connect:Direct feature, allows Processes to be submitted even if the DTF is not active. YES is the default.
FOLD = YES | NO	The FOLD parameter determines printing upper case control. It specifies whether printed output, an option of the various SELECT commands, will be all uppercase letters. The default value is NO, specifying printed output of uppercase and lowercase letters.
NODE = nodename	specifies the name of the node you want to sign on to.
PACCT ='pnode accounting data'	specifies the accounting data for the primary node (PNODE). The maximum length of the accounting data is 256 characters. If special characters, such as spaces, are part of the accounting data, you must enclose the string in parentheses. This data is the default for each Process unless you override the data on the SUBMIT command, PROCESS statement, or SUBMIT statement.
SACCT = 'snode accounting data'	specifies the accounting data for the secondary node (SNODE). The maximum length of the accounting data is 256 characters. If special characters, such as spaces, are part of the accounting data, you must enclose the string in parentheses. This data is the default for each Process unless you override the data on the SUBMIT command, PROCESS statement, or SUBMIT statement.
TMPDD = temporary data set dlblname	specifies the DLBLname of a user-supplied preallocated temporary data set (TMPDSN). The maximum length is an eight-character alphanumeric string. Do not use both the TMPDD and TMPDSN parameters at the same time in the SIGNON command.
TMPDSN = temporary data set name	specifies the name of a preallocated data set which Connect:Direct creates to receive command results that will be printed or displayed. The data set name is a 1-44 alphanumeric character string that must conform to MVS standards for data set names. Do not use both the TMPDD and TMPDSN parameters at the same time in the SIGNON command.

Using SIGNON Through Batch Interface

To use the SIGNON command from the batch interface, place your commands in a batch job stream. See *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91. If you have the Extended Submit Facility (ESF) available, the Connect:Direct DTF does not need to be running in order to submit the job.

The following example shows a SIGNON command. This example specifies accounting data for the primary and secondary nodes: PACCT for the PNODE and SACCT for the SNODE.

```
SIGNON NETMAP = DALLAS.NETMAP                                      -
   USERID = (MYUID1,MYPSWD)                                        -
   PACCT  = 'JOB FOR SYSMAINT, DEPT.  27, MARKETING'               -
   SACCT  = 'JOB FOR INVENTRY, DEPT.  55, ACCOUNTING'
SIGNOFF
```

Signing on to Multiple Sessions

The multiple-session SIGNON command allows you to sign on to multiple Connect:Direct DTFs simultaneously.

You enter the multiple-session SIGNON command through:

- Batch Interface
- Operator Interface

Refer to the *Connect:Direct Console Operator's Guide* for information on using the Operator Interface.

Command Format

The multiple-session SIGNON command has the following format and associated parameters.

Label	Command	Parameters
(optional)	SIGNON	USERID=(userid, pswd[,newpswd])
		NODE=nodename
		CASE=YES | NO
		PACCT='pnode accounting data'
		SACCT='snode accounting data'

Required Parameters

The multiple-session SIGNON requires the following parameters:

Parameter	Description
USERID = (userid, pswd [,newpswd])	specifies security information about the user who is signing on. **userid** specifies the userid being used when signing on to the DTF. It can contain 1-8 alphanumeric characters. **pswd** specifies the password associated with the userid. This optional parameter depends on your installation's security procedures. If your installation requires a password, it must be a 1-8-character alphanumeric value. **newpswd** specifies the new password being assigned to the userid. This parameter is optional. If you specify a new password, it must be a 1-8-character alphanumeric value. This action changes your password if your installation supports this option. **Note:** When you attempt a SIGNON to a remote node, the userid and password information must correspond to security procedures at that node.
NODE = nodename	specifies which Connect:Direct node to sign on to. This name must be 1-16 alphanumeric characters with the first character alphabetic.

Optional Parameters

The following parameters are optional for the multiple-session SIGNON:

Parameter	Description
CASE = YES | <u>NO</u>	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
PACCT ='pnode accounting data'	specifies the accounting data for the primary node (PNODE). The maximum length of the accounting data is 256 characters. If special characters, such as spaces, are part of the accounting data, you must enclose the string in parentheses. Connect:Direct uses this data as a default for each Process unless you override the data on the SUBMIT command, PROCESS statement, or SUBMIT statement.
SACCT = 'snode accounting data'	specifies the accounting data for the secondary node (SNODE). The maximum length of the accounting data is 256 characters. If special characters, such as spaces, are part of the accounting data, you must enclose the string in parentheses. This data is the default for each Process unless you override the data on the SUBMIT command, PROCESS statement, or SUBMIT statement.

Using Multiple-Session SIGNON Through Batch Interface

To use the multiple session SIGNON command through the batch interface, place your commands in a batch job stream, like the example presented in Chapter 6, *Using Utility Programs*, and submit the job while Connect:Direct is running. If you have the Extended Submit Facility (ESF) available, Connect:Direct DTF does not need to be running for you to submit the job.

Using one job stream, you can request that Connect:Direct functions be performed on multiple nodes by using the SIGNON, SWAP NODE, and SIGNOFF commands.

The first successful SIGNON command in the input establishes the master session. The master session is the active session until another session is established. Each subsequent SIGNON becomes the active session unless you swap sessions using the SWAP NODE command.

Signing on to the master session and signing on to another node under the master session require two different signons. The parameters specified for the master session signon dictate the environment for all sessions running under the master session. For sessions–other than master sessions–you can only use the following SIGNON parameters:

- USERID=(userid,pswd,newpswd)
- NODE=nodename
- RECONNECT

The following example shows how these parameters might be coded:

```
SIGNON   USERID=(XYZ123, SMITH, JONES)                               -
         NODE=CD.DALLAS
```

The NODE parameter, shown in the previous example, is the name assigned to a node defined in the Connect:Direct Network Map definitions.

The APPLIDS entry in the network map definitions determines the maximum number of sessions that you can sign on to during a session. Each signon uses an additional VTAM APPLID.

Note: You must carefully position SIGNOFF commands in the job stream. If you sign off the master session, then all other sessions will be signed off. If you sign off a nonmaster session, however, only that particular session will be signed off. The key to positioning SIGNOFF commands is to remember that if you sign off a nonmaster session, the master session immediately becomes the active session again. This change occurs regardless of the order in which the sessions were established.

When issuing commands in one continuous job stream, you may want one node to perform one function and another node to perform a different function. The SWAP NODE command allows you to swap from node to node. For a description of the SWAP NODE command, see *Swapping Nodes* on page 18.

Examples

The following three examples show how SIGNON, SIGNOFF, and SWAP NODE commands affect the use of multiple sessions.

Example 1–SIGNON to Two Different Nodes

Consider a SIGNON to two different nodes and the use of the SWAP NODE command. The SIGNOFF shown causes both sessions to be terminated, because the active session at SIGNOFF was the master session.

```
/* NETMAP = CD.NETMAP, BOSTON IS ACTIVE (MASTER)                   */
   SIGNON    USERID=(SAM) NODE=CD.BOSTON
/* SIGN ON TO JERSEY                                               */
   SIGNON    USERID=(SAM) NODE=CD.JERSEY
/* SUBMIT COPY ON JERSEY                                           */
   SUB   PROC=COPY1
/* SWAP TO BOSTON                                                  */
   SWAP  NODE CD.BOSTON
/* SUBMIT COPY ON BOSTON                                           */
   SUB   PROC=COPY1
/* SIGN OFF BOTH NODES (terminating master session)                */
   SIGNOFF
```

Example 2–Termination of a Master Session with SIGNOFF

Consider a SIGNOFF command issued in the job stream. In the following example, the first SIGNOFF terminates the CD.NEWYORK node because it was the last SIGNON issued. The second SIGNOFF command terminates both the CD.JERSEY and CD.BOSTON nodes because the first SIGNOFF made the master session the active session. Signing off the master session terminates all other sessions.

```
/* NETMAP = CD.NETMAP, BOSTON IS ACTIVE (MASTER)                   */
   SIGNON USERID=(SAM) NODE=CD.BOSTON
/* SIGN ON TO JERSEY                                               */
   SIGNON USERID=(SAM) NODE=CD.JERSEY
/* SIGN ON TO NEWYORK                                              */
   SIGNON USERID=(SAM) NODE=CD.NEWYORK
/* BOSTON IS ACTIVE AFTER SIGNOFF TO CD.NEWYORK                    */
   SIGNOFF
/* SIGN OFF ALL SESSIONS (terminating master session)              */
   SIGNOFF
```

Example 3–Termination of a Nonmaster Session with SIGNOFF

Consider that whenever a SIGNOFF command is issued to a nonmaster session, the master session automatically becomes active again regardless of the order in which the nodes were signed on.

If you try to swap to a node that is already active, Connect:Direct ignores the SWAP NODE command. For the node to be signed on to, the USERID (or any other user ID specified) must be one of the following:

- User record in the Connect:Direct authorization file
- Valid ID for the security subsystem in use

```
/* NETMAP = CD.NETMAP, BOSTON IS ACTIVE (MASTER                   */
   SIGNON     USERID=(SAM) NODE=CD.BOSTON
/* SIGN ON TO JERSEY                                              */
   SIGNON     USERID=(SAM) NODE=CD.JERSEY
/* SIGN ON TO NEWYORK                                             */
   SIGNON     USERID=(SAM) NODE=CD.NEWYORK
/* NEWYORK ACTIVE                                                 */
   SIGNOFF
/* SIGN ON TO JERSEY                                              */
   SWAP NODE CD.JERSEY
/* JERSEY ACTIVE                                                  */
   SIGNOFF
/* SIGN OFF ALL SESSIONS                                          */
   SIGNOFF
```

Swapping Nodes

The SWAP NODE command allows you to swap to another Connect:Direct node when you are signed on to multiple Connect:Direct nodes with the Multiple Session Signon Facility. The SWAP NODE command specifies the node to receive your command.

You enter the SWAP NODE command through:

- Batch Interface
- User-Written Applications
- Operator Interface

Refer to the *Connect:Direct Console Operator's Guide* for information on using the Operator Interface.

Command Format

The SWAP NODE command format and associated parameters follow. The required parameter appears in bold print.

Label	Command	Parameters
(optional)	SWAP NODE	nodename

Required Parameter

Parameter	Description
nodename	specifies the node to which you are swapping in a multiple-session environment.

Optional Parameters

There are no optional parameters for the SWAP NODE command.

Using SWAP NODE Through Batch Interface

To use the SWAP NODE command from the batch interface, you must place your commands in a batch job stream. Refer to the sample presented in *Using Connect:Direct in the Batch Environment* on page 89 and submit the job while Connect:Direct is running. If you have the Extended Submit Facility (ESF) available, Connect:Direct DTF does not need to be running for you to submit the job. SWAP NODE specifies which node receives your command.

The following sequence of commands swaps you from your current active local node to the CD.REMOTE node under the USERID of JONES2.

```
SIGNON USER ID=(JONES1)                                               -
         NODE NAME=CD.LOCAL
SIGNON USER ID=(JONES2)                                               -
         NODE NAME=CD.REMOTE
SWAP NODE  CD.REMOTE
SELECT NETMAP WHERE(NODE=CD.REMOTE)
SIGNOFF
SIGNOFF
```

Signing Off Connect:Direct for VSE/ESA

The SIGNOFF command terminates a connection to the Connect:Direct for VSE/ESA DTF. You can issue a SIGNOFF command that terminates your connection to Connect:Direct by any one of the following methods:

- Batch Interface
- User-Written Applications
- Operator Interface

See the *Connect:Direct Console Operator's Guide* for information on using the Operator Interface.

Command Format

The SIGNOFF command has no associated parameters.

Label	Command	Parameters
(optional)	SIGNOFF	(none)

Using SIGNOFF Through Batch Interface

To use the SIGNOFF command from the batch interface, you must place your commands in a batch job stream. Refer to the example presented in *Using Connect:Direct in the Batch Environment* on page 89 and submit the job while Connect:Direct is running. If you have the Extended Submit Facility (ESF) available, you can submit the job without Connect:Direct DTF running.

The following example shows a typical SIGNON command followed by a SIGNOFF command.

```
SIGNON   USERID=(MYUID1,MYPSWD) NETMAP=NETMAP.NAME
SIGNOFF
```

Chapter 3

Using Process Control Commands

This chapter introduces the Connect:Direct for VSE/ESA command systems and provides information about using the commands. You can use the Process Control commands to manipulate Connect:Direct Processes. This chapter details the following items:

- Writing Connect:Direct for VSE/ESA Commands
- Understanding Connect:Direct Syntax
- Entering Process Control Commands
- Submitting Processes
- Viewing Process Status
- Stopping Nonexecuting Processes
- Changing Nonexecuting Processes
- Deleting Nonexecuting Processes
- Stopping Executing Processes

See the Connect:Direct Processes Web site at

www.sterlingcommerce.com/Documentation/processes/processhome.html for information about Process statements (such as the COPY and RUN TASK statements).

Writing Connect:Direct for VSE/ESA Commands

This section describes the format and syntax you use to construct Connect:Direct for VSE/ESA commands and Processes.

Connect:Direct commands use a unique syntax that allows a continuous string of characters to convey user requests and Process statements to the Connect:Direct for VSE/ESA Data Transfer Facility (DTF) for execution.

The following table shows how the syntax for a Connect:Direct for VSE/ESA statement is documented:

Label	Command	Parameters and Subparameters

Item	**Description**
Label (optional)	Connect:Direct commands have user-chosen labels, which are character strings beginning in column 1. The label consists of 1-8 alphanumeric characters with the first character alphabetic.
Command	The command specifies the Connect:Direct function requested and must begin after column 1. Separate commands from parameters by one or more blanks or commas.
Parameters and Subparameters	Parameters and subparameters specify further instructions for the command and must be set apart by one or more blanks or commas. Parameters can be either keyword or positional. **Keyword parameters** are usually followed by an equal sign and can have a set of subparameters. **Positional parameters** must be entered in a specific order, with commas replacing any parameter not entered. These parameters are always on the right of the equal sign. Positional subparameters must be enclosed in parentheses, with the parentheses preceded and followed by blanks or commas.

Understanding Connect:Direct Syntax

This section describes the syntax used in constructing Connect:Direct command and Process statements.

Asterisk

Asterisks introduce comments. Any statement with an asterisk in column 1 is treated as a comment and the Connect:Direct command parser will ignore it.

```
*  this is a comment
```

```
* SIGNON USERID=(id,,newpswd)   SO IS THIS
```

Commas

Commas have two functions:

- Separating items in a list
- Controlling the order of values specified as positional parameters. A comma must be used to indicate omission of a positional parameter as shown in this example.

```
SIGNON USERID=(id,,newpswd)
```

Comments

Comments allow you to include additional information in a Connect:Direct for VSE/ESA Process.

Comments are available for the user's convenience and do not affect Connect:Direct. Comments are allowed in the following formats:

- Preceded by a slash-asterisk (/*) and followed by an asterisk-slash (*/), as shown in this example.

```
/* This Process copies payroll files. */
```

- Preceded by a slash-asterisk (/*), continuing over multiple lines, and terminated by an asterisk-slash (*/), as shown in this example. (The terminating */ cannot begin in column 1.)

```
/********************************************
 This step copies Florida's invoice data sets.
*********************************************/
```

- After a continuation mark, as shown in this example.

```
CHANGE PROCESS WHERE (PNAME = PAYROLL)        - This is a comment.
               DEST = DALLAS1
```

- An asterisk (*) in column 1, followed by the comment, as shown in this example.

```
* After submitting this Process,
* enable the Connect:Direct for MS-DOS node.
```

Note: For statements that will be processed by DMBATCH, only the last method of commenting is allowed. For information about DMBATCH, see *Using Connect:Direct in the Batch Environment* on page 89.

Concatenation

Concatenation links character strings to form a single string. The CONNECT:Direct concatenation operator is two vertical bars (||). The operator must be preceded and followed by a blank. For example:

```
DSN=CD || NODE
```

Resolves to:

```
DSN=CDNODE
```

Concatenation is used to:

- Join a long file name that has been continued over multiple lines.

  For example:

  ```
  DSN= $DISK1:[SMITH.LOGICALS.CD.LOGON]||                      -
       OGIN.COM;3'
  ```

  Resolves to:

  ```
  DSN='$DISK1:[SMITH.LOGICALS.CD.LOGON]LOGIN.COM;3'
  ```

  Note: This file name is enclosed in backslashes because quotation marks must be maintained as part of the string. (The DSN is part of a Process submitted from Connect:Direct for VSE/ESA to Connect:Direct for OpenVMS). See *Bracketing Characters* on page 27 for details on bracketing backslashes.

- Join Process accounting data that has been continued over multiple lines. For example:

  ```
  PACCT=JOBACCTDATA  ||                                            -
  WITHNOBLANKS
  ```

  Resolves to:

  ```
  PACCT=JOBACCTDATAWITHNOBLANKS
  ```

- Join long symbolic parameters that span multiple lines. For example:

```
&SEL=mem1,mem2,mem3,mem4,mem5,mem6,||                        -
       em7,mem8,mem9)
```

Resolves to:

```
&SEL=(mem1,mem2,mem3,mem4,mem5,mem6,mem7,mem8,mem9)
```

- Join values that are represented as symbolic parameters. For example:

```
&USERID=BOB
DSN=CD  ||  &USERID
```

Resolves to:

```
DSN = CDBOB
```

Resolution of the symbolic parameter occurs *before* concatenation.

For example, if you want to concatenate the characters XYZ and FRED to result in the string XYZFRED, do the following:

```
&ABC = FRED
PARM = XYZ  ||  &ABC
```

Do not do the following because the result will be XYZ&ABC:

```
&ABC = FRED
PARM = XYZ  ||  &AB ||  C
```

Continuation Marks

CONNECT:Direct commands can continue across multiple records. The continuation mark is the hyphen (-) as the last character on a line. There must be a blank between the last character of the statement and the hyphen indicating continuation. The continuation hyphen can appear anywhere in columns 2-80.

Parentheses

Parentheses enclose lists and associate a group of values. For example, the FROM parameter of the COPY statement is enclosed in one set of parentheses. Lists in the FROM parameter are nested in subsequent pairs of parentheses.

Single and Double Quotation Marks

Single quotation marks allow you to embed special characters or blanks in a parameter or subparameter value. You use double quotation marks in the same situations as single quotes with the exception that symbolic substitution takes place within a double-quoted string and not within a single-quoted string.

The following are examples using single and double quotation marks.

```
COPY TO (DSN='VMFILE FILETYPE')
```

```
COPY TO (DSN="C:CDIRAT.EXE")
```

Special Characters

CONNECT:Direct uses hyphens (-), double vertical bars (||), ampersands (&), and the following CONNECT:Direct delimiters as special characters:

Delimiters	Description
	blank
<	less-than sign
>	greater-than sign
*	asterisk
(	open parenthesis
)	close parenthesis
:	not sign
/	slash
	backslash
,	comma
.	period
'	single quotation mark
"	double quotation mark
=	equal sign
{	opening brace
}	closing brace
[	opening bracket
]	closing bracket

Bracketing Characters

Bracketing characters indicate special processing of a character string. Once the character string is resolved, the bracketing characters do not remain part of the string.

The CONNECT:Direct bracketing characters are:

- backslashes (
- single quotation marks (')
- double quotation marks (")

Use bracketing backslashes for clarity and consistency. Use bracketing characters:

- To continue a string containing special characters across multiple lines
- To ensure that special characters in the string, such as quotation marks, are maintained

You must code both bracketing characters on the same line. If a string containing special characters continues across multiple lines, you must enclose and concatenate each line containing a special character in bracketing characters.

For example, the following SYSOPTS parameter for Connect:Direct for VSE/ESA is a quoted string. If the string continues across multiple lines, it must be enclosed in backslashes:

```
SYSOPTS=CMD(      ||                                            -
        NDBRKMSG ||                                             -
            )"
```

Resolves to:

```
SYSOPTS="CMD(SNDBRKMSG)"
```

When you use a bracketing character that is the same as a character in the string, you must treat the special character in the string differently. To distinguish the special character from a bracketing character, double the special character in the string that you want to keep. For example:

```
PACCT=DEPTMIS   ||                                            -
              02'
```

Resolves to:

```
PACCT='DEPTIS602'
```

Symbolic Substitution

To substitute information in a CONNECT:Direct Process you can use symbolic substitution. When CONNECT:Direct encounters an ampersand (&) and 1-8 alphanumeric characters, CONNECT:Direct substitutes a string represented by the ampersand and alphanumeric characters. For example, the following can be included as part of a CONNECT:Direct Process:

```
DSN=&DATA
```

The following example uses bracketing backslashes to permit symbolic substitution in quoted strings.

PROC1 copies a data set to a file on user IVVBs 191 disk on VM. Placing bracketing backslashes around the parts of the string with special characters (in this case, the blank following **'FN**) allows resolution of the symbolic *&FILTYP*, retaining the single-quoted string to be passed to VM.

```
PROC2    PROCESS   SNODE=CD.VM &FILTYP=FT
                   COPY FROM (DSN=MVS.DATA                          -
                              DISP=SHR)                             -
                        TO   (DSN=\’FN \ &FILTYP  \’\               -
                              LINK=(IVVB,WIVVB,W,191)               -
                              DISP=(RPL))
```

Termination

A statement terminates at the end of data without a continuation mark.

Entering Process Control Commands

The Process control commands can be entered through:

- Batch Interface
- User-Written Applications
- Operator Interface

Refer to the *CONNECT:Direct Console Operator's Guide* for information on using the Operator Interface.

Submitting Processes

The SUBMIT command allows you to submit a Process for execution.

Note: Parameters specified on the SUBMIT command will override any parameters specified on the PROCESS statement.

Command Format

The SUBMIT command has the following format and associated parameters.

Label	Command	Parameters
(optional)	**SUBmit**	**PROC=member**
		DSN=mt | mt(member)
		CASE=YES | NO
		CLASS=n
		HOLD=YES | NO | CALL
		NEWNAME=newname
		PACCT='pnode accounting data'
		PNODE=primary node name
		PNODEID=(id,pswd[,newpswd])
		PRTY=n
		REQUEUE=YES | NO
		RETAIN=YES | NO | INITIAL
		SACCT='snode accounting data'
		SNODE=secondary node name | TCPNAME=tcpvalue
		SNODEID=(id,pswd[,newpswd])
		STARTT=([date | day][, hh:mm:ssXM])
		&symbolic name1=variable string1 &symbolic name2=variable string2 ... &symbolic namen=variable stringn

Required Parameters

The SUBMIT command requires the following parameters. You must specify either PROC or DSN. Do not specify both.

Parameter	Description
PROC = member	specifies the Process name. The Process name is a 1-8 character alphanumeric string, with the first character alphabetic. The Process resides in the Process Library file specified by the LIBDEF in the CONNECT:Direct JCL.

Parameter	Description
DSN = mt | mt (membername)	specifies the member type and member name in parentheses. The Process resides in the Process library specified by the LIBDEF statement in the CONNECT:Direct JCL. You must specify mt, which defaults to N, as a single character A-Z, 0-9, @, #, or $.

Optional Parameters

The following parameters are optional for the SUBMIT command.

Parameter	Description
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
CLASS = n	determines the node-to-node session on which a Process can execute.
HOLD = YES | NO | CALL	specifies whether CONNECT:Direct places the Process in the hold queue. **YES** specifies that the Process remain in the hold queue until CONNECT:Direct releases the Process. When you specify both HOLD=YES and a STARTT value, the HOLD specification takes precedence. Therefore, if you submit a Process with HOLD=YES, CONNECT:Direct places the Process in the hold queue even if you specified a start time. **NO** specifies that the Process will not be placed in the hold queue. If No is selected, CONNECT:Direct places the Process in the WAIT for execution queue (EX). HOLD=NO is the default. **CALL** specifies that CONNECT:Direct place the Process in the hold queue until a VTAM session establishes with the specified SNODE in one of the following ways: ◆ SNODE contacting the PNODE ◆ Another Process starting on the PNODE destined for the same SNODE For example, a Process submitted HOLD=NO establishes a session and executes any Processes residing on the SNODE destined for this node that were submitted with HOLD=CALL.
NEWNAME = newname	specifies the new name for the Process. The default is the label on the CONNECT:Direct Process statement. To override the default, you specify this parameter.
PACCT = 'pnode accounting data'	specifies the accounting data for the PNODE. The maximum length of the accounting data is 256 characters. If you use special characters in the accounting data, you must enclose the string in parentheses. **Note:** This data overrides any accounting data specified on the SIGNON command, and any accounting data specified in the Process statement of the submitted Process.

Parameter	**Description**
PNODE = primary node name	is a 1-16 character alphanumeric name that specifies the primary node. The name can be expressed in alphanumerics or nationals (@#$) with embedded periods. The first character must be alphabetic because it is the node that controls the Process execution. The PNODE is the submitter's node. If you specify a PNODE that is not the same as the submitter node, the PNODE will be ignored. This parameter is for documentation purposes only.
PNODEID = (id, pswd [, newpswd])	specifies security userid and security password at the PNODE. **id** specifies the 1-8-character alphanumeric security ID that passes to a security exit for validations on the PNODE side of the Process. **pswd** specifies the current security password. The security exit can use this parameter to validate the current security password on the PNODE side of the Process. **newpswd** specifies the new 1-8-character alphanumeric security password. The security exit uses this parameter to change the current security password to the new security password.
PRTY = n	specifies the Process priority in the Transmission Control Queue. See Chapter 5, *Managing Processes* for more information about the TCQ. High numbers indicate high priorities; low numbers indicate low priorities. CONNECT:Direct uses this priority only for Process selection. This Process does not affect the priority during transmission. If you do not specify PRTY, CONNECT:Direct uses the default priority defined during Connect:Direct for VSE/ESA installation. The range is from 0-15.
REQUEUE = YES | <u>NO</u>	specifies whether a COPY step should requeue if an abend occurs during processing. **YES** allows the requeued Process to be placed in the HOLD queue with a status of HELD IN ERROR (HE). Corrective action can then be taken and the Process can be restarted with the failing step. The Process must be explicitly released from the HOLD queue when the status is HELD IN ERROR (HE). **NO** specifies that the failing copy step will not be requeued if it fails with an abend. The remaining steps in the Process will execute. The default value for REQUEUE is NO.
RETAIN = YES | <u>NO</u> | INITIAL	specifies whether CONNECT:Direct is to hold a copy of the Process in the HOLD queue after the Process has executed. **YES** specifies that a copy of this Process be kept in the HOLD queue after the Process has executed. The copy of the Process will not execute until it is released by a CHANGE Process command. If you select RETAIN=Y, CONNECT:Direct automatically holds the Process until the Process is released unless the STARTT parameter is coded. Use RETAIN with STARTT to run a Process repeatedly at a given interval. **<u>NO</u>** specifies that the system delete the Process after execution. The default value for RETAIN is NO. **INITIAL** specifies that the Process be executed every time CONNECT:Direct is initialized. **Note:** Do not code STARTT with RETAIN=I.

Parameter	Description
SACCT = 'snode accounting data'	specifies the accounting data for the SNODE. The maximum length of the accounting data is 256 characters. If special characters are in the accounting data, you must enclose the string in parentheses. **Note:** This data overrides any accounting data specified on the SIGNON command, and any accounting data specified in the Process statement of the submitted Process.
SNODE = secondary node name | TCPNAME = tcpvalue	specifies the secondary node to be used in this Process. This parameter overrides the value specified in the Process and the value coded in the PROCESS statement. CONNECT:Direct allows the PNODE and the SNODE parameters to specify the same node name, to enable a PNODE = SNODE transfer. **secondary node name** is a 1-16 alphanumeric character name that is defined in the network map. The name can be expressed in alphanumerics or nationals (@#$) with embedded periods. **TCPNAME=tcpvalue** specifies TCP/IP connections that are not defined in the CONNECT:Direct network map. **tcpvalue** is defined as the TCP/IP network address, the network name, or an alias for the network name. The tcpvalue can be from 1-16 characters with embedded periods. If the network name is longer than 16 characters, you must specify an alias for the network name. If the TCPNAME keyword is used, the default TCP/IP port number is assumed. **Note:** For Interlink TCP/IP support, if tcpvalue specifies the name of the network, CONNECT:Direct will use this name to get the real address from the network name server on the TCP/IP network. Due to an IBM restriction, CONNECT:Direct does not support the conversion of name to IP address for IBM TCP/IP. For this reason, if you are using IBM TCP/IP support, you must specify the IP address in the SNODE parameter.
SNODEID = (id, pswd[, newpswd])	specifies security userid and security password at the SNODE. **id** specifies the 1-8-character alphanumeric security ID that passes to a security exit validation on the SNODE side of the Process. **pswd** specifies the current 1-8-character alphanumeric security password. The security exit validates the current security password on the SNODE side of the Process. **Note:** For CONNECT:Direct for Tandem, the IBM MVS, VM, and VSE nodes only recognize passwords specified in uppercase alphanumeric characters. Therefore, a CONNECT:Direct Process cannot be successfully initiated from MVS, VM, or VSE with CONNECT:Direct for Tandem unless the SNODEID password follows the same convention–no lowercase characters and no control characters. **newpswd** specifies the new 1-8-character alphanumeric security password. The security exit uses this parameter to change the current security password to the new security password.

Parameter	Description
STARTT = ([date | day][, hh:mm:ssXM])	specifies the date or time when the Process is to be executed. The DATE (or DAY) and TIME are positional parameters. If the DATE or DAY is not specified, a comma must precede the time. **Note:** Do not code STARTT with RETAIN=I. **date** specifies that the Process is to be held until a specific date. You can specify the day (dd), month (mm), and year (yy). Connect:Direct for VSE/ESA is Year 2000 compliant and detailed information on Year 2000 compliance is presented in the *Connect:Direct for VSE/ESA Release Notes*. To specify the order of a Gregorian day, month, and year, you *must* define the DATEFORM initialization parameter. If you do not specify the DATEFORM parameter, Connect:Direct for VSE/ESA defaults to the VSE/ESA user setup option STDOPT DATE which is specified at system initial program load (IPL). Once you have specified the order, you can use the following formats according to the order you selected: DATEFORM=MDY accepts the following formats: mmddyy *or* mmddyyyy mm/dd/yy *or* mm/dd/yyyy mm.dd.yy *or* mm.dd.yyyy DATEFORM=DMY accepts the following formats: ddmmyy *or* ddmmyyyy dd/mm/yy *or* dd/mm/yyyy dd.mm.yy *or* dd.mm.yyyy DATEFORM=YMD accepts the following formats: yymmdd *or* yyyymmdd yy/mm/dd *or* yyyy/mm/dd yy.mm.dd *or* yyyy.mm.dd DATEFORM=YDM accepts the following formats: yyddmm *or* yyyyddmm yy/dd/mm *or* yyyy/dd/mm yy.dd.mm *or* yyyy.dd.mm Connect:Direct for VSE/ESA also processes Julian dates the same as previous CONNECT:Direct for VSE releases. The following formats are valid: yyddd *or* yyyyddd yy/ddd *or* yyyy/ddd yy.ddd *or* yyyy.ddd **day** specifies the day of the week that CONNECT:Direct releases the Process for execution. Valid names are MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. If you specify the day of the week with RETAIN=Y, the Process executes the same day every week. If you only specify DAY, the time defaults to 00:00. For example, if you submit a Process on Monday, with Monday as the only STARTT parameter, the Process does not run until the following Monday. You can also specify TODAY, which releases the Process for execution TODAY; or TOMORROW, which releases the Process for execution the next day.

Parameter	Description
STARTT (*continued*)	**hh:mm:ssXM** indicates the time of day in hours (hh), minutes (mm), and seconds (ss) that the Process is released. You can set *XM* to AM or PM, or you can omit this parameter if you use a 24-hour clock. For the time of day, you can use the 24-hour clock or the 12-hour clock. Minutes and seconds are not required. If you use the 12-hour clock, 01:00:00 hours could be 1:00AM, and 13:00 hours could be 1PM. If you use the 24-hour clock, valid times are 00:00-24:00. If you do not use AM and PM, CONNECT:Direct uses the 24-hour clock. If hh:mm:ssXM is coded with RETAIN=Y, the Process executes the same time every day. You can also specify NOON, which releases the Process for execution at 12:00, or MIDNIGHT, which releases the Process for execution at 24:00. **Note:** If you specify both HOLD=YES and a STARTT value, the HOLD specification takes precedence. Therefore, CONNECT:Direct places a Process submitted with HOLD=YES in the hold queue even if you specify a start time.
&symbolic name 1 = variable string 1 &symbolic name 2 = variable string 2... &symbolic name N = variable string N	specifies the default value to be substituted for the symbolic parameter in the Process. **Note:** This value overrides any default values specified in the Process statement. You can specify a null value if a comma immediately follows the equal sign (=). You must enclose symbolic parameters containing special characters in single quotation marks, the only valid delimiter for symbolic name variable string combination. The symbolic parameter for the SUBMIT statement must begin with a single ampersand. An ampersand symbolic parameter can be set to another single ampersand symbolic parameter that was resolved during the first Process submission. Do not use identical symbolic names. The maximum variable string length is 256 characters.

Using SUBMIT Through Batch Interface

To issue the SUBMIT command through batch interface, place the command in a batch job stream like the example in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91. Submit the job while CONNECT:Direct is running. If you have the Extended Submit Facility (ESF), you can submit the job without having CONNECT:Direct DTF running.

Example 1–With Hold

The following example shows a SUBMIT command for the Process named PAYROLL. CONNECT:Direct puts this Process in the HOLD queue and retains a copy even after it releases the Process for execution.

```
SUBMIT PROC = PAYROLL                                                -
HOLD = Y  RETAIN = Y                                                 -
PACCT = '1959, DEPT 27'                                              -
SACCT = '1962, DEPT 24'
```

Example 2–With Resolution of Symbolic

The following example shows a SUBMIT command for the Process named COPYSEQ. The symbolic &DSN resolves in the Process COPYSEQ upon submission.

```
SUBMIT PROC = COPYSEQ                                                 -
NODE = CHICAGO, &DSN = MYFILE                                         -
HOLD = Y  RETAIN = Y                                                  -
PACCT = '1959, DEPT 27'                                               -
SACCT = '1962, DEPT 24'
```

Viewing Process Status

The SELECT PROCESS command allows you to examine Processes in the Transmission Control Queue. See Chapter 5, *Managing Processes* for information about the TCQ. You can specify the search criteria.

Command Format

The SELECT PROCESS command uses the following format and associated parameters.

Label	Command	Parameters
(optional)	**SELect PROCess**	**WHERE (**
		DEST=node|(list)
		PNAME=name|(list)
		PNUMber=number|(list)
		QUEUE=ALL|EXEC|HOLD|TIMER|WAIT
		STATUS=Process status|(list)
		SUBmitter=(nodename,userid)|(list)
		)
		CASE=YES | NO

Required Parameter

The CONNECT:Direct SELECT PROCESS command requires the following parameter:

Parameter	Description

WHERE (DEST = node | (list)
PNAME = name | (list)
PNUMber = number | (list)
QUEUE = ALL | EXEC | HOLD |TIMER | WAIt
STATUS = Process status | (list)
SUBmitter = (nodename, userid) | (list))
	specifies which Processes to select. You can specify the subparameters PNAME, PNUMBER, and SUBMITTER.

Optional Subparameters

These parameters are significant because they identify the Processes to be selected for the change activity defined by the other parameters.

Parameter	Description
DEST = node | (list)	is an optional subparameter which specifies the destination node names of Processes to be selected. **node** specifies the destination node which searches the Process. **(list)** specifies a list of destinations by enclosing them in parentheses and separating them by commas.
PNAME = name | (list)	specifies the names of Processes to be selected. **name** specifies that CONNECT:Direct search the Process to be selected by this Process name. **(list)** specifies a list of Processes to be selected by enclosing them in parentheses and separating them by commas.
PNUMber = number | (list)	specifies the Process numbers of Processes to be selected. **number** specifies that CONNECT:Direct search the Process to be selected by this Process number. **(list)** specifies a list of Process numbers by enclosing them in parentheses and separating them by commas. The range is 1-99,999.
QUEUE = ALL | EXEC | HOLD | TIMER | WAIT	is an optional subparameter that specifies the queue for which CONNECT:Direct searches. If QUEUE is not specified, the default is ALL. The following choices are valid: **ALL** selects Processes from all queues. **EXEC** selects Processes from the executing queue. **HOLD** selects Processes from the hold queue. **TIMER** selects Processes from the timer queue. **WAIT** selects Processes from the wait queue.

Parameter	Description
STATUS = Process status | (list)	is an optional subparameter which specifies that CONNECT:Direct search for a Process by Process status. Possible values are HC, HE, HI, HO, HP, HR, HS, RA, RH, WC, **H**, **R**, and **W**. See *TCQ Status and State Values* on page 76 for a detailed explanation of these values. The values **H**, **R**, and **W** are generic specifications for any Process status that begins with these letters. For example, if **R** is specified, Processes with RA and RH status values appear. **(list)** specifies a list of status values by enclosing them in parentheses and separating them with commas.
SUBmitter = (nodename, userid) | (list)	specifies the nodenames and userids of the people that submitted the Process or Processes. **nodename, userid** specifies that CONNECT:Direct search for the Process by the node ID and userid of the person who submitted the Process. **(list)** specifies a list of submitter IDs. You must enclose each submitter ID in parentheses and separate them from the next ID by a comma.

Optional Parameter

There is one optional parameter for the SELECT PROCESS command.

Parameter	Description
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.

Using SELECT PROCESS Through Batch Interface

To use the SELECT PROCESS command from batch interface, you must place your commands in a batch job stream, like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91, and submit the job while CONNECT:Direct is running. The following examples illustrate the format of SELECT PROCESS.

Example 1–Search by Submitter

The following command searches for a Process by submitter node ID and userid.

```
SEL PROC  WHERE (SUB = (CD.DALLAS, SMITH))
```

Example 2–Search by Process Name

The following command searches for a Process by Process name PAYROLL.

```
SEL PROC  WHERE (PNAME = PAYROLL)
```

Suspending Process Execution

The SUSPEND PROCESS command stops execution and puts an executing Process in the HOLD queue. You can restart the Process by specifying the RELEASE parameter of the CHANGE PROCESS command.

Command Format

The SUSPEND PROCESS command has the following format and associated parameters. Required parameters and keywords appear in bold print.

Label	Command	Parameters
(optional)	**SUSpend PROCess**	**WHERE (**
		PNAME=name|(list)
		PNUMber=number|(list)
		SUBmitter=(nodeid,userid)|(list)
		)
		CASE=YES | NO
		FORCE

Required Parameter

The CONNECT:Direct SUSPEND PROCESS command requires the following parameter:

Parameter	Description

WHERE (PNAME = name | (list)
PNUMber = number | (list)
SUBmitter = (nodeid, userid) | (list))
	specifies which Processes to suspend. You can specify the subparameters PNAME, PNUMBER, and SUBMITTER.

Optional Subparameters

These parameters are significant because they identify the Processes to be selected for the change activity defined by the other parameters.

Parameter	Description
PNAME = name | (list)	specifies the names of Processes to be suspended. **name** specifies that CONNECT:Direct search for the Process to be suspended by this Process name. **(list)** specifies a list of Processes to be suspended by enclosing them in parentheses and separating them by commas.
PNUMber = number | (list)	specifies the Process number(s) to be suspended. **number** specifies that CONNECT:Direct search by this Process number for the Process to be suspended. **(list)** specifies a list of Process numbers by enclosing them in parentheses and separating them by commas. The range is 1-99,999.
SUBmitter = (nodeid, userid) | (list)	specifies the node IDs and userids of the people that submitted the Process or Processes. **nodeid, userid** specifies that CONNECT:Direct search for the Process by the node ID and userid of the person who submitted the Process. **(list)** specifies a list of submitter IDs. You must enclose each submitter ID in parentheses and separate them from the next ID by a comma.

Optional Parameters

The CASE and FORCE parameters are optional on the SUSPEND PROCESS command.

Parameter	Description
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
FORCE	allows you to SUSPEND a Process that is waiting for unavailable resources. If the initial SUSPEND attempt fails, retry the command with FORCE. **Note:** If you specify FORCE for a Process executing on an LU6.2 session, this parameter may terminate the session immediately and the Process statistics will not be exchanged between nodes.

Using SUSPEND PROCESS Through Batch Interface

To use the SUSPEND Process command from batch interface, you must place your commands in a batch job stream, like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91, and submit the job while CONNECT:Direct is running.

The following examples illustrate the format of SUSPEND Process with SUBMITTER search and PNAME and PNUMBER search.

Example 1–Suspend by SUBMITTER

The following command suspends an executing Process searched for by SUBMITTER.

```
SUSPEND PROC WHERE (SUB = (DALLAS, SMITH))
```

Example 2–Suspend by PNAME and PNUMBER

The following command suspends Processes searched for by PNAME and PNUMBER.

```
SUS PROC  WHERE (PNAME = (PAYROLL,COPY100),                          -
            PNUM = (514,575))
```

Changing Nonexecuting Processes

The CHANGE PROCESS command enables you to modify a Process and its associated parameters when it is in the TCQ and in a nonexecuting state.

Command Format

The CHANGE PROCESS command has the following format and associated parameters. Required parameters and keywords appear in bold print.

Label	**Command**	**Parameters**
(optional)	**CHange PROCess**	**WHERE (**
		PNAME=name | (list)
		PNUMber=number | (list)
		SUBmitter=(nodeid,userid)|(list)
		)
		CASE=YES | NO
		CLASS=n
		DEST=destination node
		HOLD=YES | NO | CALL
		PRTY=nn

Label	Command	Parameters
		RELEASE
		RESTART=[NO|
		FIRST=volume sequence number
		FIRST=SER=volume serial number
		LAST=volume sequence number|
		LAST=SER=volume serial number |
		VOLCNT = n]
		RETAIN=YES | NO | INITIAL
		STARTT=([date | day] [,hh:mm:ssXM])

Required Parameter

The CONNECT:Direct CHANGE PROCESS command requires the following parameter:

Parameter	Description

WHERE (PNAME = name | (list)
PNUMber = number | (list)
SUBmitter = (nodename, userid) | (list))
	specifies which Process or Processes to change. You can specify the subparameters PNAME, PNUMBER, or SUBMITTER.

Optional Subparameters

These parameters are significant because they identify the Processes to be selected for the change activity defined by the other parameters.

Parameter	Description
PNAME = name | (list)	specifies the names of Processes to be changed. **name** specifies that CONNECT:Direct search the Process by this Process name. **(list)** specifies a list of Processes by enclosing them in parentheses and separating them by commas.
PNUMber = number | (list)	specifies the Process numbers of Processes to be changed. **number** specifies that the Process to be changed will be searched for by this Process number. **(list)** specifies a list of Processes by enclosing them in parentheses and separating them by commas. The range is from 1-99,999.

Parameter	Description
SUBmitter = (nodeid, userid) | (list)	specifies the node IDs and userids of the users that submitted the Processes to be changed. **nodeid, userid** specifies that CONNECT:Direct search for the Process by the node ID and userid of the person who submitted the Process. **(list)** specifies a list of submitter IDs. You must enclose each submitter ID in parentheses and separate them from the next ID by a comma.

Optional Parameters

The following parameters are optional for the CONNECT:Direct CHANGE PROCESS.

Parameter	Description
CASE = YES | <u>NO</u>	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
CLASS = n	determines the node-to-node session on which a Process executes. Consult your CONNECT:Direct administrator regarding which class to specify. The range is 1-255.
DEST = destination node	specifies a new destination node. This parameter names the node with which this Process communicates.
HOLD = YES | <u>NO</u> | CALL	specifies the relationship of the Process to the hold queue. **YES** specifies that the Process remain in the hold queue until one of the following events occurs: A CHANGE PROCESS command releases the Process for execution. A DELETE PROCESS command deletes the Process from the queue. When you specify both HOLD=YES and a STARTT value, the HOLD specification takes precedence. Therefore, if you submit a Process with HOLD=YES, CONNECT:Direct places the Process in the hold queue even if you specified a start time. **<u>NO</u>** specifies that the Process execute as soon as possible. HOLD=NO is the default. **CALL** specifies that the Process be put in the hold queue until the specified node calls in looking for work. When you specify RETAIN=Y and HOLD=N or **C**, the HOLD parameter will be ignored.

Parameter	Description
PRTY = nn	specifies a new Process priority in the TCQ. High numbers indicate high priorities; low numbers indicate low priorities. This priority is used for Process selection and does not affect the priority during transmission. The range is from 00-15.
RELEASE	specifies that the Process can be released into execution from the queue where it is currently residing. Specify RELEASE or leave off; yes and no are not valid because this is not a keyword parameter.

RESTART = [NO |
FIRST = volume sequence number |
FIRST=SER=volume serial number |
LAST = volume sequence number |
LAST = SER = volume serial number |
VOLCNT = n]
	specifies the conditions for restarting an interrupted Process.
	Use the CHANGE PROCESS command to cause CONNECT:Direct to restart a data transmission at the last checkpoint position taken before the interruption or at a previous checkpoint position. CONNECT:Direct restarts the copy step at the position you specify.
	NO specifies that the copy step should restart at the beginning of the transmission.
	FIRST=volume sequence number specifies that the copy step should restart at the beginning of the volume designated by the volume sequence number.
	FIRST=SER=volume serial number specifies that the copy step should restart at the beginning of the volume serial given.
	LAST=volume sequence number specifies that the copy step should restart at the end of the volume designated by the volume sequence number. LAST specifies that the copy step should restart at the last block on the volume if the output is disk or the last checkpoint on the volume if the output is tape.
	LAST=SER=volume serial number specifies that the copy step should restart at the end of the volume serial given. LAST specifies that the copy step should restart at the last block on the volume if the output is disk or the last checkpoint on the volume if the output is tape.
	VOLCNT=n specifies that the volume count on the interrupted copy step be changed to the value specified. This can be used to increase the number of output volumes if the copy step was interrupted because the volume count of that step was too small.

RETAIN = YES | <u>NO</u> | INITIAL	specifies whether or not CONNECT:Direct keeps a copy of the Process in the HOLD queue after the Process has executed. **YES** keeps a copy of this Process in the HOLD queue after the Process executes. The copy of the Process does not execute until you release it through a CHANGE Process command. If RETAIN=YES is specified, CONNECT:Direct automatically holds the Process until you release it, unless you include the STARTT parameter in your Process. Use RETAIN in conjunction with STARTT to cause a Process to run repeatedly at a given interval. **<u>NO</u>** specifies that CONNECT:Direct is to delete the process after execution. The default value for RETAIN is NO. **INITIAL** specifies that CONNECT:Direct is to execute the Process every time CONNECT:Direct is initialized. Do not code the STARTT parameter with the RETAIN=I parameter.

Parameter	Description
STARTT = ([date | day] [,hh:mm:ssXM])	specifies that the Process execution time is to be set or reset. The DATE, DAY, and TIME are positional parameters. If the DATE or DAY is not specified, a comma must precede the time. DAY, DATE, and TIME are handled as one unit, so a change to one will affect the other two specifications. Therefore, when changing the DATE, DAY, or TIME, you might need to respecify even the subparameters that do not change. **Note:** Do not code STARTT with RETAIN=I. **date** specifies that the Process is to be held until a specific date. You can specify the day (dd), month (mm), and year (yy). Connect:Direct for VSE/ESA is Year 2000 compliant and detailed information on Year 2000 compliance is presented in the *Connect:Direct for VSE/ESA Release Notes*. To specify the order of a Gregorian day, month, and year, you *must* define the DATEFORM initialization parameter. If you do not specify the DATEFORM parameter, Connect:Direct for VSE/ESA defaults to the VSE/ESA user setup option STDOPT DATE which is specified at system initial program load (IPL). Once you have specified the order, you can use the following formats according to the order you selected: DATEFORM=MDY accepts the following formats: mmddyy *or* mmddyyyy mm/dd/yy *or* mm/dd/yyyy mm.dd.yy *or* mm.dd.yyyy DATEFORM=DMY accepts the following formats: ddmmyy *or* ddmmyyyy dd/mm/yy *or* dd/mm/yyyy dd.mm.yy *or* dd.mm.yyyy DATEFORM=YMD accepts the following formats: yymmdd *or* yyyymmdd yy/mm/dd *or* yyyy/mm/dd yy.mm.dd *or* yyyy.mm.dd DATEFORM=YDM accepts the following formats: yyddmm *or* yyyyddmm yy/dd/mm *or* yyyy/dd/mm yy.dd.mm *or* yyyy.dd.mm Connect:Direct for VSE/ESA also processes Julian dates the same as previous CONNECT:Direct for VSE releases. The following formats are valid: yyddd *or* yyyyddd yy/ddd *or* yyyy/ddd yy.ddd *or* yyyy.ddd

Parameter	Description
STARTT (*continued*)	For additional information about Connect:Direct for VSE/ESA compliance with Year 2000 requirements, refer to the *Connect:Direct for VSE/ESA Release Notes*. **Note:** If RETAIN=Y, a date cannot be specified in the STARTT parameter. **day** specifies the day of the week that the Process is to be released for execution. Valid names include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. If you specified the day of the week with RETAIN=Y, the Process will execute the same day every week. If only day is specified, the time will default to 00:00. This means that if a Process is submitted on Monday, with Monday as the only STARTT parameter, the Process will not run until the following Monday. You can also specify TODAY, which will release the Process for execution today, or TOMORROW, which will release the Process for execution the next day. **hh:mm:ssXM** indicates the time of day in hours (hh), minutes (mm), and seconds (ss) that the Process is to be released. *XM* can be set to AM or PM. The time of day can be expressed using the 24-hour clock or the 12-hour clock. If you use the 24-hour clock, valid times are 00:00 to 24:00. If the 12-hour clock is used, 1:00 hours would be expressed as 1:00AM, and 13:00 hours would be expressed as 1PM. If AM and PM are not used, the 24-hour clock is assumed. If you specified hh:mm:ssXM with RETAIN=Y, the Process would execute at the same time every day. You also can specify NOON, which will release the Process for execution at noon, or MIDNIGHT, which will release the Process for execution at midnight.

Using CHANGE PROCESS Through Batch Interface

To use the CHANGE PROCESS command from batch interface, place your commands in a batch job stream, like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91, and submit the job while CONNECT:Direct is running.

The following examples illustrate the use of CHANGE Process with HOLD=Y, PNAME search, and PNUM search.

Example 1–With HOLD=Y

In the following example, a list of two Processes is to be placed in the hold queue.

```
CH PROC  WHERE (PNAME = (PAYROLL, BILLING))                         -
          HOLD = Y
```

Example 2–Search by PNAME

In this example, CONNECT:Direct searched the Process by PNAME, the new destination node is DALLAS1, and the Process executes every Friday.

```
CH PROC  WHERE (PNAME = PAYROLL                                    -
          DEST = DALLAS1                                           -
          RETAIN = Y                                               -
          STARTT = (FR))
```

Example 3–Search by PNUMBER

In the following example, CONNECT:Direct searched the Process by PNUMBER. Because the RETAIN status was changed, the Process will be deleted after execution.

```
CH PROC WHERE (PNUM = 60584)                                        -
         RELEASE                                                    -
         RETAIN = N
```

Deleting Nonexecuting Processes

The DELETE PROCESS command allows you to specify the name of a nonexecuting Process and remove it from the TCQ. By way of comparison, note that the FLUSH Process command stops an executing Process.

Command Format

The DELETE PROCESS command has the following format and associated parameters. Required parameters and keywords appear in bold print.

Label	Command	Parameters
(optional)	**DELete PROCess**	**WHERE (**
		PNAME=name | (list)
		PNUMber=number | (list)
		SUBmitter=(nodeid,userid)|(list)
		)
		CASE=YES | NO

Required Parameters

The CONNECT:Direct DELETE PROCESS command requires the following parameter:

Parameter	Description

WHERE (PNAME = name | (list)
PNUMber = number | (list)
SUBmitter = (nodeid, userid) | (list))
	specifies which Processes to delete. You can specify the subparameters PNAME, PNUMBER, or SUBMITTER.

Optional Subparameters

These parameters are significant because they identify the Processes to be selected for the change activity defined by the other parameters.

Parameter	**Description**
PNAME = name | (list)	specifies the names of Processes to be deleted. **name** specifies the Process names to be deleted. **(list)** specifies a list of Processes by enclosing them in parentheses and separating them by commas.
PNUMber = number | (list)	specifies the Process numbers of Processes to be deleted. **number** specifies the Process number to be deleted. **(list)** specifies a list of Process numbers by enclosing them in parentheses and separating them by commas. The range is from 1-99,999.
SUBmitter = (nodeid, userid) | (list)	specifies the node IDs and userids of the users that submitted the Processes to be changed. **nodeid, userid** specifies that CONNECT:Direct search for the Process to be deleted by the node ID and userid of the person who submitted the Process. **(list)** specifies a list of submitter IDs. You must enclose each submitter ID in parentheses and separate them from the next ID by a comma.

Optional Parameter

There is one optional parameter for the DELETE PROCESS command.

Parameter	**Description**
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.

Using DELETE PROCESS Through Batch Interface

To use the DELETE PROCESS command from batch interface, place your commands in a batch job stream, like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91, and submit the job while CONNECT:Direct is running.

The following examples illustrate the format of DELETE Process with PNAME search, PNUMBER search, and SUBMITTER search.

Batch DELETE Process Searched by PNAME

In this example, CONNECT:Direct searches the Process by PNAME and then deletes it.

```
DEL PROC WHERE (PNAME = PAYROLL)
```

Example 1–Search by PNUMBER

In this example, CONNECT:Direct searches the Process by PNUMBER, and then deletes it.

```
DEL PROC WHERE (PNUM = 60584)
```

Example 2–Search by SUBMITTER

In this example, CONNECT:Direct searches the Process by SUBMITTER, and then deletes it.

```
DEL PROC WHERE (SUB = (DALLAS.VSE, CHUCK))
```

Stopping Executing Processes

The FLUSH PROCESS command terminates an executing Process. Note that if the Process is specified with RETAIN=YES, the flushed Process will remain on the TCQ for execution at the next scheduled time. If RETAIN=NO, the Process will be removed from the TCQ and must be resubmitted if you want it to execute.

Command Format

The FLUSH PROCESS command has the following format and associated parameters. Required parameters and keywords appear in bold print.

Label	Command	Parameters
(optional)	**FLUSH PROCess**	**WHERE (**
		PNAME=name | (list)
		PNUMber=number | (list)
		SUBmitter=(nodeid,userid) | (list)
		)
		CASE=YES | NO
		FORCE

Required Parameters

The CONNECT:Direct FLUSH PROCESS command requires:

Parameter	Description

WHERE (PNAME = name | (list)
PNUMber = number | (list)
SUBmitter = (nodeid, userid) | (list))
	specifies which Processes to flush. You can specify the subparameters PNAME, PNUMBER, or SUBMITTER.

Optional Subparameters

When you select two or more of these qualifiers, join them with AND. These parameters are significant because they identify the Processes to be selected for the change activity defined by the other parameters.

Parameter	Description
PNAME = name | (list)	specifies the names of Processes to be flushed. **name** specifies that CONNECT:Direct search by this Process name for the Processes to be flushed. **(list)** specifies a list of Processes by enclosing them in parentheses and separating them by commas.

Parameter	Description
PNUMber = number | (list)	specifies the Process numbers of Processes to be flushed. **number** specifies that CONNECT:Direct search the Process to be flushed by this Process number. **(list)** specifies a list of Processes by enclosing the Process numbers in parentheses and separating them by commas. The range is from 1-99,999.
SUBmitter = (nodeid, userid) | (list)	specifies the node IDs and userids of the users that submitted the Processes to be flushed. **nodeid, userid** specifies that CONNECT:Direct search for the Process to be flushed by the node ID and userid of the person who submitted the Process. **(list)** specifies a list of submitter IDs. You must enclose each submitter ID in parentheses and separate them from the next ID by a comma.

Optional Parameters

The following parameters are optional for the FLUSH PROCess command.

Parameter	Description
CASE = YES | NO	determines case sensitivity. It specifies whether parameters associated with accounting data, userid, password, and data set name in the commands which follow the signon are to be case sensitive. The default is NO. You can override this designation on a command-by-command basis by adjusting the case sensitivity parameter on the individual command.
FORCE	allows you to SUSPEND a Process that is waiting for unavailable resources. If the initial SUSPEND attempt fails, retry the command with FORCE. **Note:** If you specify FORCE for a Process executing on an LU6.2 session, this parameter may terminate the session immediately and the Process statistics will not be exchanged between nodes.

Using FLUSH PROCESS Through Batch Interface

To use the FLUSH PROCESS command from batch interface, place your commands in a batch job stream, like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91, and submit the job while CONNECT:Direct is running.

The following examples illustrate the format of FLUSH Process with SUBMITTER search, PNAME search and PNUMBER search.

Example 1–Search by SUBMITTER

The following command flushes an executing Process searched by SUBMITTER.

```
FLUSH PROC WHERE (SUB = (DALLAS, SMITH))
```

Example 2–Search by PNAME and PNUMBER

The following command flushes Processes searched by PNAME and PNUMBER.

```
FLUSH PROC WHERE (PNAME = (PAYROLL,COPY100),                         -
         PNUM = (514,575))
```

Chapter 4

Using Environment Commands

This chapter details Connect:Direct environment commands and provides information about using the commands. This chapter presents the following commands:

- SELECT STATISTICS
- SELECT NETMAP
- SELECT TYPE
- STOP CD
- MODIFY

Viewing Statistics

Connect:Direct for VSE/ESA keeps a record of all operations and activity in the Statistics Log file. You can use the SELECT STATISTICS command to retrieve and examine statistic log records such as Process Submission, Process Termination, COPY Termination, RUNJOB, RUNTASK, or Connect:Direct Console messages.

You can enter the SELECT STATISTICS command through:

- Batch Interface
- Operator Interface

Refer to the *Connect:Direct Console Operator's Guide* for information on the operator interface.

Command Format

The SELECT STATISTICS command has the following format and associated parameters. The required parameters and keywords appear in bold print.

Label	Command	Parameters
(optional)	SELect STATistics	WHERE (
		CCODE=(condition,completion code)
		EXCLUDE=(MEMBer|MCR)|(WTO)|(NOTWTO)| (list)
		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
		ARCHDSN=dsname | (list)
		FILE | PRint | TABle | SUMmary
		)

Required Parameter

The following is a required parameter for the Connect:Direct SELECT STATISTICS command.

Parameter	Description

WHERE (CCODE = (condition,completion code)
EXCLUDE = (MEMBer | MCR) | (WTO) |
(NOTWTO)|(list)
PNAME = name | (list)
PNUMber = number | (list)
STARTT = ([date | day] [,hh:mm:ssXM])
STOPT = ([date | day] [,hh:mm:ssXM]))
	specifies which statistics record(s) you want to examine. You must specify at least one of the subparameters, PNAME, PNUMber, or STARTT.

Optional Subparameters

The following parameters are optional for the Connect:Direct SELECT STATISTICS command.

Parameter	Description
CCODE = (condition, completion code)	specifies that Connect:Direct search the statistics records 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), or LE (less than or equal to). **completion code** specifies a completion code value, which ranges from 1-2,147,483,647, to allow for all values passed by a RUN TASK, RUN JOB, or COPY statement. This last value represents a maximum 31-bit binary number. For example, if CCODE = (GT,0) is specified, you will see statistics records in which the step completion code is greater than zero, as long as the records also meet other specified criteria.
EXCLUDE = (MEMBer | MCR) | (WTO) | (NOTWTO) | (list)	excludes certain statistics from selection. Specify a list of excluded options by enclosing them in parentheses and separating them by a space or a comma. To select all statistics, omit the EXCLUDE parameter from the SELECT STATISTICS command. The TYPE parameter overrides the EXCLUDE specifications. **MEMBer** | **MCR** specifies whether or not to exclude the PDS member copy record for PDS copies on the statistics report. **WTO** specifies that Connect:Direct write-to-operator (WTO) messages be excluded from the SELECT STATISTICS command. Dynamic allocation messages are represented as WTOs in the statistics file. **NOTWTO** specifies that only WTO records be displayed (exclude everything that is NOT a WTO record). **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.
PNAME = name | (list)	specifies the names of the Processes to be searched. You can specify a list of Processes by enclosing them in parentheses. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.
PNUMber = number | (list)	specifies the Process Connect:Direct searches for by Process number. You can specify a list of Processes by enclosing them in parentheses. The range is from 1-99,999. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.

Parameter	Description
STARTT = ([date | day] [,hh:mm:ssXM])	specifies that Connect:Direct search for statistics records by a designated starting date and time. The DATE, DAY, and TIME are positional parameters. If you do not specify the DATE or DAY, a comma must precede the time. **date** specifies the date from which the statistics records will be selected. You can specify the day (dd), month (mm), and year (yy).Connect:Direct for VSE/ESA is Year 2000 compliant and detailed information on Year 2000 compliance is presented in the *Connect:Direct for VSE/ESA Release Notes*. To specify the order of a Gregorian day, month, and year, you *must* define the DATEFORM initialization parameter. If you do not specify the DATEFORM parameter, Connect:Direct for VSE/ESA defaults to the VSE/ESA user setup option STDOPT DATE which is specified at system initial program load (IPL). Once you have specified the order, you can use the following formats according to the order you selected: DATEFORM=MDY accepts the following formats: mmddyy *or* mmddyyyy mm/dd/yy *or* mm/dd/yyyy mm.dd.yy *or* mm.dd.yyyy DATEFORM=DMY accepts the following formats: ddmmyy *or* ddmmyyyy dd/mm/yy *or* dd/mm/yyyy dd.mm.yy *or* dd.mm.yyyy ddmmyy *or* ddmmyyyy yymmdd *or* yyyymmdd yy/mm/dd *or* yyyy/mm/dd yy.mm.dd *or* yyyy.mm.dd DATEFORM=YDM accepts the following formats: yyddmm *or* yyyyddmm yy/dd/mm *or* yyyy/dd/mm yy.dd.mm *or* yyyy.dd.mm Connect:Direct for VSE/ESA also processes Julian dates the same as previous Connect:Direct for VSE releases. The following formats are valid: yyddd *or* yyyyddd yy/ddd *or* yyyy/ddd yy.ddd *or* yyyy.ddd If you only specify the date, the time defaults to 00:00. **day** specifies the day of the week for which Connect:Direct searches the statistics records. Valid names include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. You can also specify YESTER or TODAY. YESTER or YESTERDA searches for statistics records created yesterday. TODAY searches for the statistics records created today.

Parameter	Description
STARTT (*continued*)	**hh:mm:ssXM** indicates the time of day in hours (hh), minutes (mm), and seconds (ss) for which Connect:Direct searches the statistics records. You can set XM to AM or PM. You can use the 24-hour clock or the 12-hour clock to express the time of day. If you use the 24-hour clock, valid times are 00:00-24:00. If you use the 12-hour clock, 1:00 hours could be expressed as 1:00AM, and 13:00 hours could be expressed as 1PM. If you do not indicate AM or PM, Connect:Direct assumes the 24-hour clock. You do not need to specify minutes and seconds. You can also specify NOON, which searches for the statistics records at 12:00, or MIDNIGHT, which searches for the statistics records at 24:00. The default for the time is 00:00:00, the beginning of the day. If you do not specify the STARTT parameter, the search criterion consists of any time before the STOPT parameter.
STOPT = ([date | day] [,hh:mm:ssXM])	specifies that Connect:Direct search statistics records for up to and including the designated DATE, DAY and TIME positional parameters. If you do not specify the DATE or DAY, you must precede the time with a comma. **date** specifies the date from which the statistics records will be selected. You can specify the day (dd), month (mm), and year (yy).Connect:Direct for VSE/ESA is Year 2000 compliant and detailed information on Year 2000 compliance is presented in the *Connect:Direct for VSE/ESA Release Notes*. To specify the order of a Gregorian day, month, and year, you *must* define the DATEFORM initialization parameter. If you do not specify the DATEFORM parameter, Connect:Direct for VSE/ESA defaults to the VSE/ESA user setup option STDOPT DATE which is specified at system initial program load (IPL). Once you have specified the order, you can use the following formats according to the order you selected: DATEFORM=MDY accepts the following formats: mmddyy *or* mmddyyyy mm/dd/yy *or* mm/dd/yyyy mm.dd.yy *or* mm.dd.yyyy DATEFORM=DMY accepts the following formats: ddmmyy *or* ddmmyyyy dd/mm/yy *or* dd/mm/yyyy dd.mm.yy *or* dd.mm.yyyy DATEFORM=YMD accepts the following formats: yymmdd *or* yyyymmdd yy/mm/dd *or* yyyy/mm/dd yy.mm.dd *or* yyyy.mm.dd

Parameter	Description
STOPT (*continued*)	DATEFORM=YDM accepts the following formats: yyddmm *or* yyyyddmm yy/dd/mm *or* yyyy/dd/mm yy.dd.mm *or* yyyy.dd.mm Connect:Direct for VSE/ESA also processes Julian dates the same as previous Connect:Direct for VSE releases. The following formats are valid: yyddd *or* yyyyddd yy/ddd *or* yyyy/ddd yy.ddd *or* yyyy.ddd For additional information about Connect:Direct for VSE/ESA compliance with Year 2000 requirements, refer to the *Connect:Direct for VSE/ESA Release Notes*. If you specify only the date, the time defaults to 00:00. **day** specifies the day of the week for which Connect:Direct searches the statistics records. Valid names include MOnday, TUesday, WEdnesday, THursday, FRiday, SAturday, and SUnday. You can also specify TODAY, which searches for the statistics records today. **hh:mm:ssXM** indicates the time of day in hours (hh), minutes (mm), and seconds (ss) for which Connect:Direct selects the statistics records. You can set XM to AM or PM. You can use the 24-hour clock or the 12-hour clock to express the time of day. If you use the 24-hour clock, valid times are 00:00-24:00. If you use the 12-hour clock, 1:00 hours could be expressed as 1:00AM, and 13:00 hours could be expressed as 1PM. You can also specify NOON, which searches for the statistics records ending at 12:00, or MIDNIGHT, which searches for the statistics records ending at 24:00. The default for the time is 24:00:00, the end of the day. If you do not specify the STOPT parameter, Connect:Direct selects the statistics until the present. If you specify only the time of day (not date) in the STARTT parameter, note that the output will show the first available entry in the statistics log for that time of day. Statistics from all later times and dates will be output up to and including the stop time. If you specify only the time of day (not date) in the STOPT parameter, all statistics through the most recent entry for that time of day will be output.

Parameter	Description
USER = name | (list)	limits the selected statistics records to those that were written for users with the specified name. You can specify a list of names by enclosing them in parentheses. Use generic specifications by placing an asterisk (*) at the end of the name. For example, if you specify USER = SYS$*, then records with SYS$ in the first four characters of the USER field are selected regardless of the contents of the remaining characters. Records having SYSBOB, SYSADM, and SYS$0001 all satisfy this selection criterion. USER names can be up to 64 characters in length and can contain lowercase characters. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.
SNODE = name | (list)	limits the selected statistics records to those that were written for Processes where the specified node name acted as SNODE. You can specify a list of names by enclosing them in parentheses. Use generic specifications by placing an asterisk (*) at the end of the name. For example, if you specify SNODE=DALLAS*, then records with SYS$ in the first four characters of the USER field are selected regardless of the contents of the remaining characters. Records having DALLAS.PROD, DALLAS.TEST, and DALLAS all satisfy this selection criterion. SNODE names can contain lowercase characters. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.

Parameter	Description
TYPE = ID | (list)	specifies the statistics record types to select. Every statistics record 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. The TYPE subparameter specifies which record types to select, and *overrides the EXCLUDE parameter*. When TYPE is not specified, the record types selected are determined by the output option chosen, and can be affected by the EXCLUDE subparameter. When the PRINT or TABLE output option is specified, the types in the following table are selected. Specify FILE for the output option to select all available types. Use the EXCLUDE subparameter to exclude certain types that the output option included. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank. **ID** **Description** PS Process Submit CT Copy Termination MC PDS Member Copy RT Run Task RJ Run Job SW Submit Within a Process WTO WTO PT Process Termination See the *Optional Connect:Direct Exits* chapter in the *Connect:Direct for VSE/ESA Administration Guide* for a list of all statistics record type identifiers.
FNAME=dsname | (list)	limits the selected statistics records to those that contain the specified filename. The FNAME subparameter is meaningful for the following record types: ◆ CT Copy Termination ◆ RJ Run Job ◆ SD Start Connect:Direct ◆ SW Submit Within a Process 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 64 characters in length and can contain lowercase characters. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.

Parameter	Description
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 purpose of the SELECT STATISTICS command. **YES** folds 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 within the session defaults if nothing is specified.
ARCHDSN=dsname | (list)	specifies that Connect:Direct is to search the archived statistics files named, *instead of the statistics log currently in use by the DTF*. Archived statistics files are used to store old statistics data that is no longer active in the statistics log. See your system administrator for information about the availability of archived statistics at your site, and for the data set names you can specify with the ARCHDSN subparameter. See *Using SELECT STATISTICS Through Batch Interface* on page 62 information on archiving statistics. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.

Optional Parameter

The following parameters are the optional parameters for the SELECT STATISTICS command.

Parameter	Description
FILE | PRint | TABle | SUMmary	The FILE, PRINT, TABLE, or SUMMARY parameter specifies the output format. Indicate only one. TABLE is the default. The **FILE** parameter causes Connect:Direct to pass back the output as unformatted records to the temporary file. You indicate the name of your temporary file through the TMPDSN parameter of your SIGNON command. See page 12 for a description of the TMPDSN parameter. The format of the record is found in the Connect:Direct sample macro library. Once Connect:Direct puts the unformatted records in a temporary file, other programs can process the records and customize reports for specific needs. This is recommended for processing of statistics records. You can archive a statistics record each day by means of a Generation Data Group (GDG). Connect:Direct provides sample job streams that extract statistics on a daily basis. The **PRint** parameter routes output of the command to the destination specified in the PRINT keyword of the SIGNON command. The **TABle** parameter stores the output of the command in the Connect:Direct temporary file and displays it upon successful completion of the command. The output is usually in tabular format. The **SUMmary** parameter produces a three-line summary per statistics record.

Using SELECT STATISTICS Through Batch Interface

To use the SELECT STATISTICS command from the batch interface, place your commands in a batch jobstream like the example presented in *Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface* on page 91 and submit the job while Connect:Direct is running.

Example 1–Search by STARTT and STOPT

The following command searches for statistics records based on start and stop time. Connect:Direct prints the output.

```
SEL STAT WHERE (                                                   -
                 EXCLUDE = (WTO)                                   -
                 STARTT = (01/24/97,06:45:00)                      -
                 STOPT = (01/25/97,8AM)                            -
                )
         PRINT
```

The preceding command results in the following output:

```
------------------------------------------------------------------------------------
CD.NYC               Connect:Direct SELECT STATISTICS               DATE: 01/25/1997
------------------------------------------------------------------------------------
  Function            =>PROCESS SUBMIT              Start Time        =>06:01:58
  Process Name        =>PAYROLL Stop Time           =>06:01:58
  Process Num         =>12      Comp Code           =>0000000
  Userid              =>JONES   Comp Msg            =>SSPA001I
  Secondary           =>        Step Name           =>
------------------------------------------------------------------------------------
Function            =>MEMBER  Start Time          =>06:02:03
Process Name        =>PAYROLL Stop Time           =>06:02:06
Process Num         =>12      Comp Code           =>0000000
Userid              =>JONES   Comp Msg            =>SCPA000I
Primary             =>CD.DALLAS                  Step Name         =>STEP1
   Name: From       =>XYZ     To                 =>ABC
         recs       =>0       blks                =>113
------------------------------------------------------------------------------------
Function            =>COPY                       Start Time        =>06:01:58
Process Name        =>PAYROLL                    Stop Time         =>06:02:07
Process Num         =>12                         Comp Code         =>0000000
Userid              =>JONES                      Comp Msg          =>SCPA000I
Primary             =>CD.DALLAS                  Step Name         =>STEP1
From (Pnode
       Dsn=TECHRPT.FILE)
          recs      =>0                         blks               =>113
      I/O Bytes     =>351,360
     VTAM Bytes     =>351,586
     Cmpr Perc      =>-0.1%
       VOL=SER      =>WK0004
To (Snode
     Dsn=NEWREPT.FILE
          recs      =>0                         blks               =>113
      I/O Bytes     =>351,360
     VTAM Bytes     =>351,586
      Cmpr Perc     =>-0.1%
        VOL=SER     =>WK0007
------------------------------------------------------------------------------------
```

Example 2–Search by STARTT and STOPT

The following command searches for statistics records based on start and stop time. Connect:Direct displays the output on the screen.

```
SEL STAT WHERE (                                        -
                  EXCLUDE = (WTO,MEMB)                  -
                  STARTT = (01/21/97,15:25:00)          -
                  STOPT = (01/25/97,15:30:00)           -
                )
```

The preceding command results in the following output:

```
------------------------------------------------------------------------------------
CD.NYC               Connect:Direct SELECT STATISTICS               DATE: 01/21/1997
------------------------------------------------------------------------------------
Function            =>PROCESS SUBMIT             Start Time         =>15:27:19
Process Name        =>ORDERS                     Stop Time          =>15:27:19
Process Num         =>143                        Comp Code          =>0000000
Userid              =>JAY                        Comp Msg           =>SSPA001I
Secondary           =>CD.CHI                     Step Name          =>
------------------------------------------------------------------------------------
Function           =>RUN JOB                     Start Time        =>15:27:22
Process Name       =>PAYROLL                     Stop Time         =>15:27:23
Process Num        =>143                         Comp Code         =>0000000
Userid             =>JAY                         Comp Msg          =>SRJA000I
Secondary          =>CD.CHI                      Step Name         =>RJ1
-----------------------------------------------------------------------------------
Function             =>PROCESS DURATION          Start Time        =>15:27:20
Process Name         =>ORDERS                    Stop Time         =>15:27:23
Process Num          =>143                       Comp Code         =>0000000
Userid               =>JAY                       Comp Msg          =>SVTM100I
Secondary            =>CD.CHI
------------------------------------------------------------------------------------
```

Example 3–Search by PNUMBER

The following command searches for statistics records based on the Process number. Connect:Direct prints the output.

```
SEL STAT WHERE (                             -
                EXCLUDE = (WTO)              -
                PNUM = 179                   -
                )
```

The preceding command results in the following output:

```
------------------------------------------------------------------------------------
CD.NYC               Connect:Direct SELECT STATISTICS               DATE: 01/21/1997
------------------------------------------------------------------------------------
Function            =>PROCESS SUBMIT             Start Time        =>17:30:56
Process Name       =>PROC01                      Stop Time         =>17:30:56
Process Num        =>179                         Comp Code         =>0000000
Userid             =>RAY                         Comp Msg          =>SSPA001I
Secondary          =>CD.DALLAS                   Step Name         =>
------------------------------------------------------------------------------------
Function           =>RUN TASK                    Start Time        =>17:30:56
Process Name       =>PROC01                      Stop Time         =>17:30:59
Process Num        =>179                         Comp Code         =>0000000
Userid             =>RAY                         Comp Msg          =>SRTA000I
Secondary           =>CD.DALLAS                   Step Name         =>STEP03
------------------------------------------------------------------------------------
Function             =>PROCESS DURATION           Start Time        =>15:27:20
Process Name         =>PROC01                     Stop Time         =>15:27:23
Process Num          =>179                        Comp Code         =>0000000
Userid               =>RAY                        Comp Msg          =>SVTM100I
Secondary             =>CD.DALLAS
------------------------------------------------------------------------------------
```

Example 4–Search by PNAME, STARTT, STOPT

The following command searches for statistics records based on Process name, start time, and stop time. Connect:Direct prints the output.

```
SEL STAT WHERE (                                   -
                EXCLUDE = (WTO)                    -
                PNAME = BILLING                    -
                STARTT = (02/28/97,06:00)          -
                STOPT = (02/28/97,23:00)           -
                )
```

The preceding command generates a detailed report with the SELECT STATISTICS parameters.

```
------------------------------------------------------------------------------------
CD.BOSTON            Connect:Direct SELECT STATISTICSDATE: 02/28/1997
------------------------------------------------------------------------------------
Function            =>PROCESS SUBMIT             Start Time         =>10:48:54
Process Name        =>BILLING                      Stop Time        =>10:48:55
Process Num         =>126                         Comp Code         =>0000000
Userid              =>JOE                         Comp Msg          =>SSPA001I
Secondary            =>CD.SANFRAN                 Step Name        =>
------------------------------------------------------------------------------------
Function            =>COPY                        Start Time        =>10:48:46
Process Name        =>BILLING                     Stop Time         =>10:48:57
ProcessComp Code         =>0000000
Userid              =>JOE                         Comp Msg          =>SCPA000I
Secondary            =>CD.SANFRAN                 Step Name         =>CP1
From (Pnode
     Dsn=A411.TEST2)
         recs       =>0                          blks              =>2
    I/O Bytes       =>1,210
   VTAM Bytes       =>1,214
    Cmpr Perc       =>-0.1%
To (Snode
     Dsn=NEWREPT.FILE
         recs       =>0                          blks              =>2
    I/O Bytes       =>1,210
   VTAM Bytes       =>1,214
    Cmpr Perc       =>-0.3%
------------------------------------------------------------------------------------
Function            =>SUBMIT W/IN PROC            Start Time        =>10:48:57
Process Name        =>BILLING                     Stop Time         =>10:48:58
Process Num         =>126                         Comp Code         =>0000000
Userid              =>JOE                         Comp Msg          =>SSUB0001
Secondary           =>CD.SANFRAN
Submit Node         =>CD.BOSTON
------------------------------------------------------------------------------------
Function            =>PROCESS DURATION            Start Time        =>10:48:55
Process Name        =>BILLING                     Stop Time         =>10:48:58
Process Num         =>126                         Comp Code         =>0000000
Userid              =>JOE                         Comp Msg          =>SVTM000I
Secondary           =>CD.SANFRAN
------------------------------------------------------------------------------------
```

Parameters Generated from Batch Interface Commands

The following paragraphs describe the parameters generated from the batch interface commands:

- PROCESS SUBMIT record
  - The PROCESS SUBMIT record provides statistics about the submitted Process.
  - The START TIME indicates when the Process was submitted.
  - The STOP TIME indicates when the SUBMIT completed.
  - The COMP CODE contains the SUBMIT completion code.
  - The COMP MSG contains the SUBMIT completion message.

Note: If you specify EXCLUDE = (MEMB) or EXCLUDE = (MCR), the member copy record would not appear on the statistics report.

- COPY termination record
  - The COPY termination record tells you how the COPY statement in BILLING was completed.
  - The START TIME indicates when the copy step started.
  - The STOP TIME indicates when the step ended. The elapsed time between START TIME and STOP TIME in the copy termination record includes file allocation on both nodes as well as the time required to mount a tape, when required.
  - The STEP NAME is the label name specified on the COPY statement. If a label is not specified, this field is left blank.
  - The RECS and BLKS fields indicate the number of records or blocks read from or written to the file and identifies whether the transfer was performed in block or record mode.
  - The I/O BYTES fields indicate the number of bytes that were read or written from disk or tape.
  - The VTAM BYTES fields indicate the number of bytes sent or received during the session, including Connect:Direct and VTAM control information.
  - The CMPR PERC field indicates the compression percentage. This percentage is calculated by dividing the number of VTAM bytes by the number of I/O bytes. The compression percentage values shown in the COPY termination record differ depending on the file attributes specified. A negative compression percentage means that more bytes were sent than were read from the file. This includes Connect:Direct control information.
- PROCESS DURATION (Process termination) record
  - The PROCESS DURATION record indicates how the whole Process was completed. Connect:Direct writes this record when the Process ends.
  - The time between the START TIME and the STOP TIME indicates the elapsed time for all steps in the Process.

  - The COMP CODE field indicates the highest completion code for all steps within the Process.
  - The COMP MSG contains the Process completion message.
- RUN JOB record
  - The RUN JOB record provides statistics about a RUN JOB statement. The RUN JOB record provides statistics about a job submitted to the POWER reader queue.
  - The time that elapsed between the START TIME and the STOP TIME indicates the time it took to read the job and submit it to the internal reader.
  - The COMP CODE includes the completion code of the submission to the internal reader.
  - The STEP NAME indicates the label, if one was specified. If you do not specify a label on the RUN JOB statement, this field will be blank.
- RUN TASK record
  - The RUN TASK record provides statistics about a program attached to the Process as a subtask.
  - The time that elapsed between the START TIME and the STOP TIME indicates the time it took to run the task before control returned to Connect:Direct.
  - The COMP CODE includes the final completion code of the task.
  - The STEP NAME is the label on the Run Task statement. If you do not specify a label (as in this case), this field will be blank.
  - SECONDARY indicates the other node involved in the Process. This field can be either secondary nodename or primary nodename.
- SUBMIT W/IN PROC (Submit statement within a Process) record
  - The SUBMIT W/IN PROC record provides statistics about a Process that was submitted within another Process.
  - The time that elapsed between the START TIME and STOP TIME indicates the time it took to submit the Process to the TCQ.
  - The COMP CODE contains the completion code of this SUBMIT statement. This code will be nonzero if the Process did not submit successfully.
  - SUBMIT NODE indicates the node on which the Process was to run.

Retrieving Network Map Records

The SELECT NETMAP command retrieves records from the netmap file for displaying or further processing. You enter the SELECT NETMAP command through:

- Batch Interface
- Operator Interface

Refer to the *Connect:Direct Console Operator's Guide* for information on using the Operator interface.

Command Format

The SELECT NETMAP command uses the following format and associated parameters. The required parameter appears in bold print. Default values for parameters and subparameters are underlined.

Label	Command	Parameters
(optional)	SELect NETmap	**WHERE** (
		NODE=node|generic|(list)
		)

Required Parameter

Connect:Direct requires the following parameter for the SELECT NETMAP command.

Parameter	Description
WHERE (NODE=(node | generic | (list))	specifies which network map node definition(s). **node** specifies a nodename. The node is a 1-16 character alphanumeric string, with the first character alphabetic. **generic** specifies a generic selection of node name(s). To specify node names generically, type a 1-7 character alphanumeric string, with the first character alphabetic, plus an asterisk (*). For instance, if your network includes node names PHOENIX, SANDIEGO, SANFRAN, and TUCSON, a specification of **SAN*** would provide information about the SANDIEGO and SANFRAN nodes. If only an (*) is specified, the system will display (or print) all nodes that you are authorized to use. **(list)** specifies multiple node names. A list of node names can be specified by enclosing them in parentheses and separating each by a comma or a blank.

Optional Parameters

There are no optional parameters for SELECT NETMAP.

Using SELECT NETMAP Through Batch Interface

If a Process submitted for execution on the DALLAS node from the SANDIEGO node does not execute, the command produces the following output:

```
         SIGNON
SNMSTEP  SEL NET     WHERE (NODE = (SAN*))
         SIGNOFF
```

The following report is issued to determine whether the user can communicate with that node:

```
===========================================================
                 SELECT NETWORK MAP
===========================================================
Node Name   : SANFRAN             VTAM Applid   : CDAPPL1
Max Parsess : 8                   Def Ses Class : 2
Session Type: SNA                 Environment   :
Node Status : INTERNAL, SEND, RECEIVE
```

In this example, the only node displayed is SANFRAN. Its VTAM APPLID is CDAPPL1, the maximum number of parallel sessions for SANFRAN is **8**, and its default session class is **2**. The display does not include information on the SANDIEGO node, so you can conclude that the Dallas node will not communicate with the San Diego node.

Chapter 5

Managing Processes

This chapter provides information about the following:

- Transmission control queue (TCQ)
- Process selection
- TCQ status and state values
- Process recovery and the Copy Checkpoint/Restart procedure

Transmission Control Queue

Processes are stored in the transmission control queue (TCQ) after submission though the SUBMIT command. The TCQ is used to control Process execution as Connect:Direct operates.

The TCQ consists of two VSAM relative record data sets (RRDS) and an in-memory queue, which controls access. The two data sets are the TCQ and the TCX, which is a space map for the TCQ. These files are inseparable.

TCQ Logical Queues

The TCQ is divided into four logical queues: Wait, Execution, Hold, and Timer.

The queues can be accessed and Processes can be manipulated through the following Connect:Direct commands that are described in Chapter 4, *Using Environment Commands*:

- CHANGE PROCESS
- DELETE PROCESS
- FLUSH PROCESS
- SELECT PROCESS
- SUSPEND PROCESS

Process Routing

The RETAIN, HOLD, and STARTT parameters cause Processes to be routed in the way described in the following table:

Parameters	Queue	Comments
None	Wait	Process remains on the Wait queue until Connect:Direct can start a session with the SNODE at which time it moves to the Execution queue.
RETAIN=INITIAL	Hold	Process will automatically execute each time Connect:Direct is initialized with TCQ=WARM.
RETAIN=YES	Hold	A copy of the Process is kept in the Hold queue after it has executed. The Process will not execute again until it is released by a CHANGE PROCESS command.
HOLD=YES	Hold	Process will remain in the Hold queue until someone releases the Process.
HOLD=CALL	Hold	Process is automatically moved from the Hold queue to the Wait queue when the SNODE contacts the node on which the Process resides.
STARTT	Timer	When the scheduled time and date arrive, the Process is put on the wait queue and is available for execution.

RETAIN=INITIAL is useful for Processes that will contact other Connect:Direct nodes each time Connect:Direct completes initialization. This will cause any work queued on the remote node for this node to begin.

RETAIN=YES when combined with STARTT can be used to run a Process at a periodic interval. For example, RETAIN=YES and STARTT=(Tuesday, 3pm) will start the Process every Tuesday at 3 pm; RETAIN=YES and STARTT=(,12:00) will start the Process each day at noon.

The Timer queue also is used for session retry and file allocation retry based on Connect:Direct initialization parameters specified by a particular installation. Once retry limits have been exhausted, the Process is moved to the Hold queue with an HE status.

Process Selection

When a Process is submitted, it is put on the appropriate logical queue based on Process statement parameters that affect scheduling. These parameters are:

- RETAIN=YES|NO|INITIAL
- HOLD=YES|NO|CALL
- STARTT=((date|day)(,hh:mm:ssXM))

The parameters are explained on the Connect:Direct Processes site at www.sterlingcommerce.com/Documentation/processes/processhome.html and in *Using SELECT PROCESS Through Batch Interface* on page 37.

Processes are selected in a first-in first-out (FIFO) manner for execution within process class and priority as sessions are available. The following section provides information on the method that Connect:Direct uses to select Processes for execution.

Planning for Parallel Sessions

Connect:Direct uses the parallel sessions capability of VTAM so that multiple Processes can execute simultaneously between any two Connect:Direct nodes. Process selection for each parallel session is based on a class the user specifies on a Process.

Note: Parallel sessions support requires specification of PARSESS=YES in the VTAM application definition for both nodes. If two nodes have differing values for parallel sessions, transfers are limited by the maximum number of sessions in the ADJACENT.NODE definition on the node where the process was submitted.

The maximum number of sessions between two nodes is defined in the network map. Because each session has a corresponding class value, the maximum number of sessions and maximum number of classes are equal. Selection of a Process for execution in a given node is based on Process priority (PRTY= on the Process statement) within session class. User-specified class values allow a Process to execute on the session having the matching class value or on sessions with higher class values.

The default class is the value specified in the Connect:Direct network map.

When one Process completes, Connect:Direct selects another Process (if any) to run on the available session. Connect:Direct searches until it finds the first Process with a class eligible to execute on the available session.

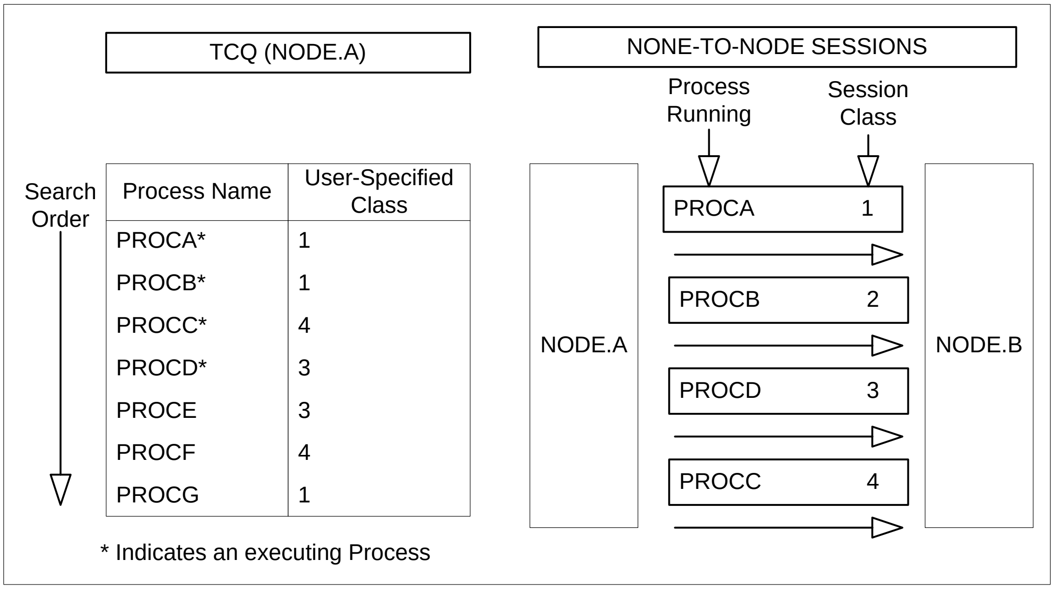
A typical use for classes is to define critical Processes with low class values so that more sessions are available for their execution. Higher class numbers can be specified for time-consuming Processes; this allows sessions with corresponding lower class numbers to become available more frequently. An example of selection by class follows.

Example of Parallel Processes

Connect:Direct has just been brought up. Seven Processes have been submitted for NODE.B and are ready to run. All Processes have a user-specified class value and the same priority. Class determines which session Connect:Direct will select.

The network map has been defined so a maximum of four sessions can be started between NODE.A and NODE.B. Each session between NODE.A and NODE.B has its own corresponding unique class number.

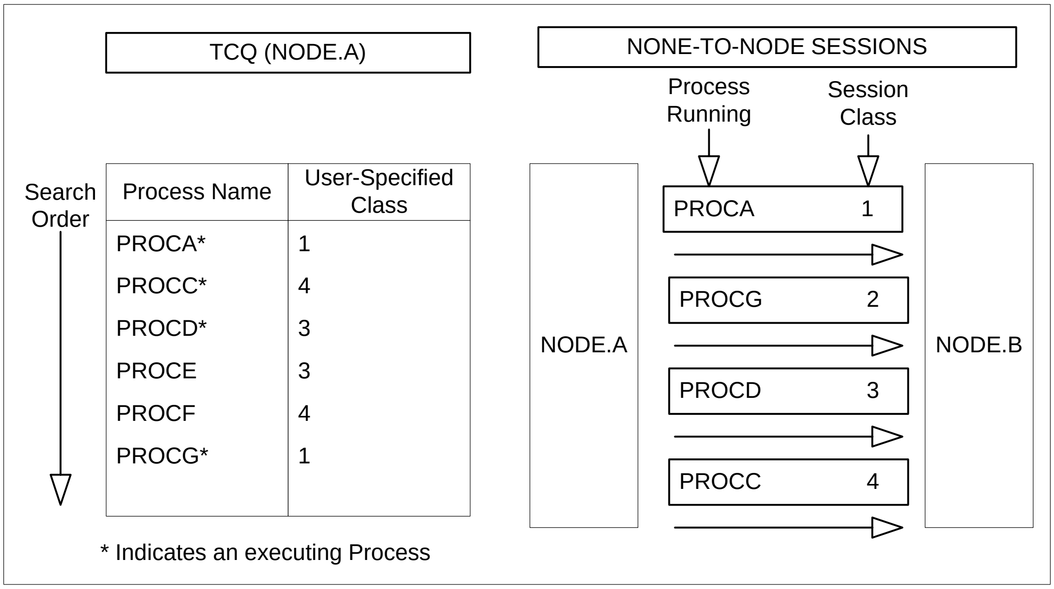
1. In the following figure, NODE.A simultaneously starts four sessions. Processes are shown in the order they appear in the queue. Note that because PROCB's user-specified class is 1, it can run on the class 2, 3, or 4 sessions if needed.



Process Name	User-Specified Class
PROCA*	1
PROCB*	1
PROCC*	4
PROCD*	3
PROCE	3
PROCF	4
PROCG	1

* Indicates an executing Process

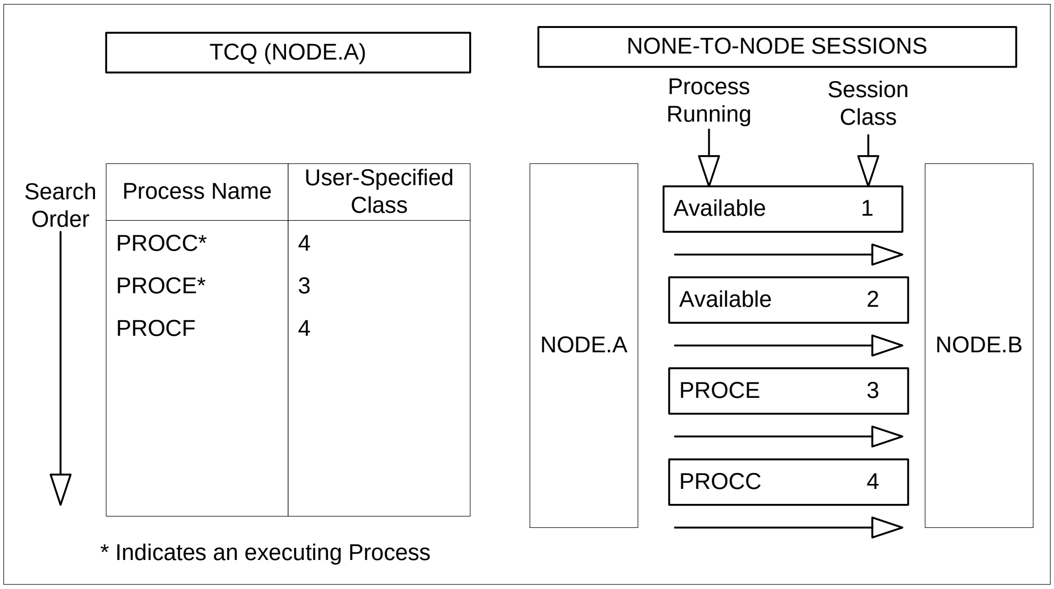
2. In the figure in step 2, PROCB has completed execution, making a session available. Connect:Direct looks through the TCQ for the first eligible Process for that session. Note that PROCG is the next Process available to run on the class 2 session because all other Processes have a class value higher than 2.



Process Name	User-Specified Class
PROCA*	1
PROCC*	4
PROCD*	3
PROCE	3
PROCF	4
PROCG*	1

* Indicates an executing Process

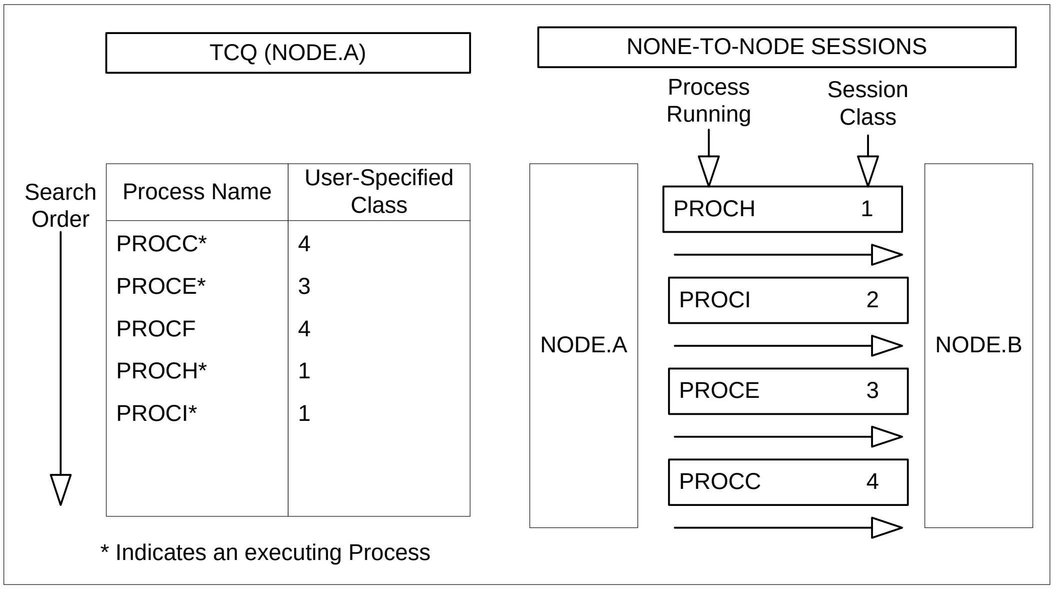
3. In the figure in step 3, PROCA, PROCG, and PROCD have completed execution. Note that sessions for class 1 and 2 cannot be used at this time because only a Process for class 4 is left in the queue.



Process Name	User-Specified Class
PROCC*	4
PROCE*	3
PROCF	4

* Indicates an executing Process

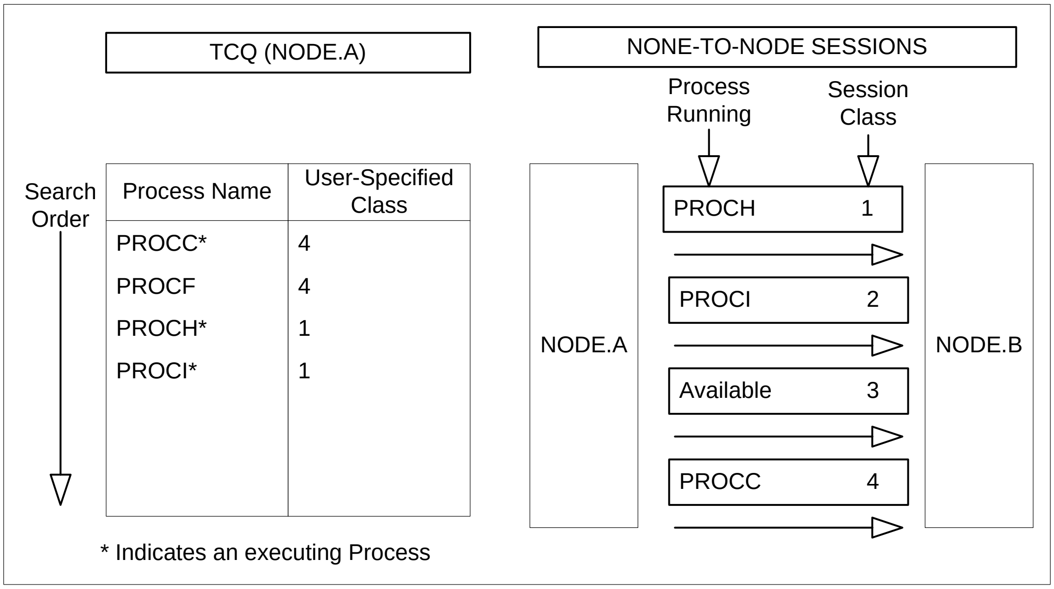
4. In the figure in step 4, PROCH and PROCI are submitted.



Process Name	User-Specified Class
PROCC*	4
PROCE*	3
PROCF	4
PROCH*	1
PROCI*	1

* Indicates an executing Process

5. In the figure in step 5, PROCE completes. Note that PROCF will not execute until PROCC has completed because PROCF can run only on the class 4 session. If at this point any other Process is submitted for class 1, 2, or 3 it can use a class 3 session.



TCQ Status and State Values

The SELECT PROCESS command displays Connect:Direct status values, Connect:Direct task state values, and VTAM state values.

Value	Description
Connect:Direct Status Value	Each Process on the TCQ has an associated Connect:Direct status value. This status value has a unique meaning determined by which queue the Process is in. The SELECT PROCESS command displays Connect:Direct status values.
Connect:Direct Task State Values	When a Process is in the Execution queue, the SELECT PROCESS command also displays a Connect:Direct task state value. The state values are provided for information purposes and cannot be modified or controlled with Connect:Direct commands. The task state value shows the current state of the Process. Usually, Connect:Direct tasks are waiting for completion of a service such as File I/O, Connect:Direct locked resource, or VTAM I/O.

Value	Description
VTAM State Values	When a Process is on the Execution queue, the SELECT PROCESS command also can display, depending on timing, a VTAM state value. The state values are provided for information purposes and cannot be modified or controlled with Connect:Direct commands. If the Process is currently in a VTAM I/O state, the SELECT PROCESS output shows a further VTAM state value such as send or receive. While a Process is executing, the SELECT PROCESS command displays the number of file blocks or records and VTAM request/response units (RUs) sent or received. This gives you an indication of the status of COPY statements on the Process.

The following tables describe the Connect:Direct status values, Connect:Direct task state values, VTAM state values, and Connect:Direct commands applicable to each of the queues.

WAIT QUEUE

The following tables describe the Connect:Direct status values and Connect:Direct commands applicable to the Wait Queue.

Connect:Direct Status Values for the Wait Queue

The following table shows the Connect:Direct Status Values for the Wait Queue.

Status Value	Explanation
WC (Waiting for Connection)	This is the initial queue status when a Process is submitted without HOLD or RETAIN specified. This status means the Process is ready to execute as soon as possible. Process is ready to run but other Processes are executing with the same SNODE, and no other sessions are available. This Process will run as soon as an eligible session is available. If you find a Process in this state, it is most likely for this reason.

Applicable Commands for the Wait Queue

The following table shows the applicable commands for the Wait Queue.

Applicable Command	Description
Change Process	Modifies Process attributes
Delete Process	Removes the Process from the queue
Select Process	Displays Process status and state

EXECUTION QUEUE

The following tables describe the Connect:Direct status values, Connect:Direct task state values, Subtask state values, VTAM state values, allocate state values, and Connect:Direct commands applicable to the Execution Queue.

Connect:Direct Status Values for the Execution Queue

The following table shows the Connect:Direct status values for the Execution Queue.

Status Value	Explanation
EXEC	Node is in Process control and executing the Process shown.
WC (Waiting for Connection)	Process control is in negotiation while the two nodes determine which Process executes next, based on priority.
:PR.CNTL	This node is not in Process control. This can occur: (1) during Process negotiation where highest priority on either node will run next, or (2) when this is the SNODE during Process execution.
SS	Session with other node is being started.

Connect:Direct Task State Values for the Execution Queue

The following table shows the Connect:Direct task state values for the Execution Queue.

Task State	Explanation
DISPATCH	Task is waiting to be dispatched
INACTIVE	Task is dispatchable but inactive
VTAM I/O	Task is waiting on VTAM request
P=SNODE	PNODE equals SNODE task
VSAM I/O	Waiting on VSAM I/O request
MISC	Miscellaneous I/O, such as a WTO
FILE I/O	Non-VSAM I/O
LOCK	Waiting for Connect:Direct locked resource
SUBTASK	Waiting on a subtask, such as open or close, allocation, security or a RUN TASK program
RUNNING	Executing instructions
TCA SCAN	TCA scan
TIMER	Waiting for timer event
LU1PRINT	Waiting for LU1 printer

Task State	Explanation
ATTACH	Waiting for tape drive to be attached
ALLOCATE	Waiting for allocation to complete
MOUNT	Waiting for tape mount
OPEN	Waiting for OPEN to complete
$	Subtask running
+	Waiting on SYNC/4 submit
@	Waiting on STATS open/close
!	Waiting for STimer to be set
#	Waiting on strings
2	Task has subtask running
WTOR	Outstanding WTOR
STR WAIT	Waiting on VSAM string
LEV2 SUB	RUN TASK attached and running

Subtask State Values for the Execution Queue

The following table shows the subtask state values for the execution queue. The first characters of the subtask request state indicate the session protocol such as TCP or 6.2.

Subtask State	Explanation
ADOPT V2	Performing Adopt
CALL V2	Performing Call
INIT V2	Performing Initialization
CLEANUP	Performing Cleanup
HANGUP	Performing Hangup
SEND V2	Performing Send
SEND RSP	Performing Send Response
SEND SIG	Performing Send Signal
RECV V2	Performing Receive
ANSWER V2	Performing Answer
IOCTL V2	Performing I/O Control
GET BUF	Performing I/O Control Get Buffer

Subtask State	Explanation
RDY RCV	Performing I/O Control Ready Receive

VTAM State Values for the Execution Queue

The following table shows the VTAM state values for the execution queue.

VTAM State	Explanation
NO SESSION	No VTAM session
SESSION EST	Session with another node being established
NO REQUEST	No VTAM request outstanding
RECEIVE	Waiting on VTAM RECEIVE request
OPEN	Waiting on VTAM OPEN request
CLOSE	Waiting on VTAM CLOSE request
SETLOGON	Waiting on VTAM SETLOGON request
REQSESS	Waiting on a request session request
OPNDST	Waiting on an open destination request
CLSDST	Waiting on a close destination request
GENCB EXTLST	Waiting on a GENCB EXLIST
GENCB ACB	Waiting on a GENCB ACB
GENCB NIB	Waiting on a GENCB NIB
GENCB RPL	Waiting on a GENCB RPL
REJSESS	Waiting on a reject session request
SESSIONC	Waiting on a session cancel request
INQUIRE	Waiting on an inquire request
OPNSEC	Waiting on an open secondary request
RSHUTD	Waiting on a request shutdown request
SIMLOGON	Waiting on a simulate logon request
SND RESPONSE	Waiting on a send response request
WAIT FOR +DR	Waiting on a definite response request
VTAM I/O	Waiting on VTAM request
SEND	Waiting on VTAM send request

VTAM State	Explanation
DACTSESS	Waiting on LU6.2 deactivate session request
RCVFMH5	Waiting on LU6.2 receive FMH-5 request
REJECT	Waiting on LU6.2 reject conversation request
ACTSESS	Waiting on LU6.2 activate session request
ALLOC ALLOCD	Waiting on LU6.2 allocate conversation until available request
PRERECV	Waiting on LU6.2 prepare to receive request
CNOS	Waiting on LU6.2 change number of session request
DELLOC	Waiting on LU6.2 deallocate conversation request
SEND CONFRMD	Waiting on LU6.2 send confirmation request
6.2 RECEIVE	Waiting on LU6.2 receive request
6.2 SEND	Waiting on LU6.2 send request
SEND ERROR	Waiting on LU6.2 send error request
DISPLAY	Waiting on LU6.2 display session limit request
DEFINE	Waiting on LU6.2 define session limits request
ALLOC IMMED	Waiting on LU6.2 allocate conversation immediately request

Allocate State Values for the Execution Queue

The following table shows the allocate state values for the execution queue.

Allocate State Value	Explanation
SVC99	Performing SVC99
CAT SEARCH1	CAMLST locate
READ VTOC	CAMLST search

Applicable Commands for the Execution Queue

The following table shows the applicable command descriptions for the execution queue.

Applicable Command	Description
Flush Process	Terminates and deletes an executing Process
Select Process	Displays Process status and state, for example

Applicable Command	Description
Suspend Process	Terminates and moves an executing Process on the hold queue

HOLD QUEUE

The following tables describe the Connect:Direct status values and Connect:Direct commands applicable to the hold queue.

Connect:Direct Status Values for the Hold Queue

The following table shows the Connect:Direct status values for the hold queue.

Status Value	Explanation
HC (Held for Call)	The Process was submitted with HOLD=CALL specified. A session started from either node will cause this Process to be put on the wait queue in WC status, and eventually EX Q when the first Process finishes.
HE (Held in Error)	The Process was submitted and received an error unrelated to allocation or session errors.
HI (Held Initially)	Process was submitted with HOLD=YES specified.
HO (Held by Operator)	An exception response was sent from the other node during FMH exchanges at process negotiation or step termination. It can also occur if a FMH is invalid or is sent out of sync, or if the remote node is not defined in the network map.
HP (Held due to Process Error)	An exception response was sent from the other node during EXIT exchanges at process negotiation or step termination. HP can also occur if an EXIT is invalid, sent out of sync, or if the remote node is not defined in the Network map.
HR (Held Retain)	Process was submitted with RETAIN=YES specified.
HS (Held for Suspension)	The operator issued a SUSPEND PROCess command. The Process can be released later.
RA (Held for Restart Due to Allocation Error)	During Process execution, an allocation error occurred that matched those specified in the initialization parameters. This allows the Process to be restarted after the allocation problem is resolved.
RH (Restart Held)	A checkpointed Process was executing when an error such as a lost session or an I/O error occurred. This allows the copy to be restarted when the session is re-established.
WC (Wait For Connection)	Session establishment was attempted, including retries if specified, and failed. This Process will be put on the wait queue (and later EX Q) if a session with that node is established later. It also can be released.

Applicable Commands for the Hold Queue

The following table shows the applicable commands for the Hold Queue.

Applicable Command	Description
Change Process	Modifies Process attributes
Delete Process	Removes the Process from the queue
Select Process	Displays Process status and state, for example

TIMER QUEUE

The following tables describe the Connect:Direct status values and Connect:Direct commands applicable to the timer queue.

Connect:Direct Status Values for the Timer Queue

The following table shows the Connect:Direct Status values for the timer queue.

Status Value	Explanation
RE (Retry)	The session with the SNODE is in the retry state. The number of and interval between retries is specified in the initialization parameters. The Process can be in retry status for session establishment or for an allocation error.
WC (Wait For Connection)	The Process was submitted with a start time or date (STARTT) that has not expired. When the STARTT is reached, the Process will be put on the wait queue for scheduling to the EX Q.

Applicable Commands for the Timer Queue

The following table shows the applicable commands for the timer queue.

Applicable Command	Description
Change Process	Modifies Process attributes
Delete Process	Removes the Process from the queue
Select Process	Displays Process status and state, for example

Process Recovery and Copy Checkpoint/Restart

Connect:Direct provides facilities to recover from most errors that occur during Process execution. Recovery from the point of failure usually can be accomplished quickly. The following types of errors can occur during normal operation:

- Link failure terminates a session between Connect:Direct systems
- File I/O error occurs during Process execution
- Connect:Direct abends because of a hardware or other error

Connect:Direct offers these facilities to address the errors:

- Session establishment retry
- Process step checkpoint
- VTAM automatic session retry
- COPY statement checkpoint/restart

The following paragraphs discuss these recovery facilities.

Session Establishment Retry

When one or more Processes are ready to run with a node, Connect:Direct will establish a session with that node and begin execution. If the session cannot be started, Connect:Direct will retry the session establishment. The number of retries and the interval between retries are specified by the MAXRETRIES and WTRETRIES initialization parameters.

If a session cannot be established after all retries are exhausted, the Processes will be placed in the TCQ on the hold queue with a status of waiting for connection (WC). When a session is established with the other node by submitting another Process, by releasing a Process being held, or by contact initiated by the remote node, all other Processes will be scanned and the highest priority Process will be executed after the previous Process is finished.

Process Step Checkpoint

As a Process executes, Connect:Direct records which step is executing in the TCQ. If Process execution is interrupted for any reason, the Process is held in the TCQ. When the Process is available for execution again, Connect:Direct will automatically begin execution at that step.

Connect:Direct always performs Process step checkpoint, which does not require any user specification.

VTAM Automatic Session Retry

If Process execution is interrupted because of a VTAM session failure, Connect:Direct will automatically attempt to restart the session. This recovery facility is similar to and uses the same parameter values as the session establishment retry facility.

If the session cannot be reestablished, the Process that was in execution and any other Processes that are ready to run with the other node will be placed in the hold queue with a status of waiting for connection (WC).

COPY Statement Checkpoint/Restart

For physical sequential (PS) files and partitioned data sets (PDS), Connect:Direct collects positioning checkpoint information at specified intervals as a COPY statement executes. Checkpoints are taken for each member that is transferred within a PDS regardless of the checkpoint interval. If the copying procedure is interrupted for any reason, it can be restarted at the last checkpoint position.

The COPY statement checkpoint/restart works in conjunction with step restart. The restart is automatic if Connect:Direct can reestablish a session based on the initialization parameter values for MAXRETRIES and WTRETRIES.

The CHANGE PROCESS command can also invoke the checkpoint/restart facility. See *Using CHANGE PROCESS Through Batch Interface* on page 45 for instructions on how to use the CHANGE PROCESS command.

Initialization Parameters

Initialization parameters, the checkpoint/restart file, and the COPY statement checkpoint parameter are elements of the checkpoint/restart facility.

Connect:Direct provides four initialization parameters used with the COPY statement checkpoint/restart facility: CKPT.MODE, CKPT, CKPT.DAYS, and CKPTDSN. See the *Connect:Direct for VSE/ESA Installation Guide* for the detailed discussion of these parameters.

Checkpoint File

The Connect:Direct checkpoint/restart file contains positioning information for both files involved in executing a COPY statement. The checkpoint records are maintained throughout data transmission and are deleted when a transmission completes successfully.

Note: Checkpoints are taken on the receiving end of a transfer. During restart, this information is exchanged with the sender so that proper positioning can take place.

A checkpoint record can be left in the checkpoint file if the operator deletes an interrupted Process. Connect:Direct will scan the checkpoint records during initialization and delete records older than the value specified in the CKPT.DAYS initialization parameter.

Additional Information Sent with Checkpoint/Restart for Non-PDS Files

Except for transfers using TCP/IP, when Checkpoint/Restart is in use, Connect:Direct will send an additional 7 bytes per block or ten bytes per record of overhead during data transfer. Data is sent in record mode when reblocking is taking place during transfer (source and destination block sizes differ). When no reblocking is taking place, Connect:Direct sends in block mode.

In the case of transfers using TCP/IP, Connect:Direct sends approximately 30 bytes of overhead after sending an amount of data equal to the checkpoint interval in effect.

COPY Statement CKPT Parameter

The CKPT parameter on the COPY statement can be used to specify the interval Connect:Direct will use to record checkpoint information. Connect:Direct uses this CKPT value, rounded to the nearest block, in determining how many bytes to transfer before taking a checkpoint. If the COPY statement CKPT parameter is not specified, Connect:Direct uses the value specified in the CKPT initialization parameter. Specifying a value of CKPT=0K or not specifying CKPT in the initialization parameters disables checkpointing.

Checkpoint/Restart Examples for LU0 Transfers

When checkpointing is in effect, positioning information is transferred in addition to the file. This information is used to reposition the file in the event of an interruption. The following examples show how to determine this overhead in number of bytes when using Checkpoint/Restart for LU0 transfers.

Example A

In the following table, Connect:Direct will checkpoint the file in block mode and add 7 bytes to each block transmitted. This will add 2,100 bytes to the transmission, or less than .04 percent overall.

File Attributes	Sending File	Receiving File
BLKSIZE	19,069	19,069
LRECL	0	0
DSORG	PS	PS
RECFM	U	U
FILE SIZE	300 blocks	300 blocks

Example B

In the following table, Connect:Direct will checkpoint the file in record mode because the data is being reblocked. Connect:Direct will add 10 bytes to each record being transmitted. This will add 150,000 bytes to the transmission, or 5 percent overall.

File Attributes	Sending File	Receiving File
BLKSIZE	5,000	2,000
LRECL	200	200
DSORG	PS	PS
RECFM	FB	FB
FILE SIZE	600 blocks	1,500 blocks

Checkpoint/Restart Example for Version 2 Transfers

Using the transfer described in the preceding table, the file size is 3 million bytes. If the checkpoint interval is 10k, 30 bytes of overhead is sent 300 times, resulting in a total overhead of 9000 bytes. This is about 0.3 percent overhead overall.

Process Recovery and RUN TASK Checkpoint/Restart

Connect:Direct for VSE/ESA offers a checkpoint/restart feature with the RUN TASK Process statement.

If a RUN TASK program is executing on the SNODE and a session failure occurs, the PNODE recognizes the session failure and puts the Process in the Timer queue for retry. The SNODE, however, is still running the RUN TASK program and is not aware of the session failure until the program finishes. The checkpoint/restart feature for RUN TASK ensures that when the Process restarts on the PNODE, the RUN TASK program will not execute a second time on the SNODE.

When and Where Checkpoint Records Are Written

Connect:Direct will always write a checkpoint record on the node where the RUN TASK program executes. The initial checkpoint record is written upon entry to DMGRUNT, the module that handles the RUN TASK Process statement. Connect:Direct updates the checkpoint record before attaching the program that is to execute. When the RUN TASK program finishes, Connect:Direct updates the checkpoint record again.

Restart Procedure

If a RUN TASK Process step restarts, the node where the program executes attempts to find the checkpoint record in the checkpoint file. If the RUN TASK step is still executing, the Process that is running for the restart of the step will wait for the RUN TASK program to finish the first task and then proceed to the next step of the Process, if there is one.

At Process restart for a RUN TASK step, if the program is still executing, you will see two Processes in the EX queue for the same Process step. The first Process is executing the program. The second Process is waiting for the first Process to complete. When the first Process completes, it determines that the session it was running under has been lost and will post the second Process. The second Process records how the RUN TASK step that was still executing ended and will proceed to the next step in the Process.

Determining Reexecution of the RUN TASK Step

The RUN TASK step will not execute again if it is determined at restart that the RUN TASK step ended because it finished before the PNODE restarted the Process. However, if the RUN TASK program did not complete and is not currently running, then the RESTART parameter determines the restart of the Process.

Also, if at restart, Connect:Direct cannot find the checkpoint record and the RUN TASK program is not executing, Connect:Direct is unable to determine what action to take for the restart. If Connect:Direct cannot determine what action to take for the restart, it will use the RESTART parameter.

You can code the RESTART parameter on the RUN TASK step or in the initialization parameters. Refer to the *Connect:Direct for VSE/ESA Installation Guide* for information on the RUNTASK.RESTART initialization parameter. If you code the RESTART parameter on the RUN TASK step, it will override the initialization parameter. When you code RESTART=YES, Connect:Direct executes the program again. When you code RESTART=NO, the Process skips the RUN TASK step.

Chapter 6

Using Utility Programs

Connect:Direct provides several utility programs that facilitate the use of Connect:Direct for VSE/ESA. This chapter presents the following items:

- DMBATCH–Batch Interface Program
- DMNOTIFY–User Notification Programs
- DMRTAMS–VSAM AMS Interface Program
- DMRTLIBR–Dynamically Run LIBR from Within a Process
- DMRTPOWR–VSE/POWER Interface Program

Using Connect:Direct in the Batch Environment

Connect:Direct provides an interface that allows you to request Connect:Direct services in the batch environment. You request the batch interface program, called DMBATCH, by submitting a job stream that contains the Connect:Direct commands as input. DMBATCH reads the input data stream and processes the requested functions. All Connect:Direct commands and command options are supported by this interface.

Batch Interface Requirements

When using the batch interface, you must meet the following three requirements:

- All Processes being submitted must reside in one of the source libraries in the LIBDEF chain.
- Process statements themselves cannot be used in the job stream. However, you can use a SUBMIT command to submit a Process that contains PROCESS statements.

Note: DMBATCH processing of the SUBMIT is asynchronous. The batch job will be notified whether the SUBMIT was successful or not, but not whether the COPY itself was successful or not.

- The Connect:Direct commands used in the job stream must follow the syntax outlined in the following section.

Batch Interface Syntax

Use the following syntax rules when you are creating the SYSIN for the job stream.

- Because DMBATCH recognizes anything in column 1 of the input as a label, all commands must begin after column 1.
- Any command requiring more than one line for completion must be furnished with a continuation mark (-) on each line that will be continued. The continuation mark can be placed anywhere on the line as long as it is preceded and followed by a blank.
- An asterisk (*) in column 1 indicates a comment.

Batch Interface Processing

The following are Batch Interface rules for entering Connect:Direct commands and options in the job stream.

- The first command in the job stream must be a SIGNON command, or no further processing will take place.
- Each new Connect:Direct command in the job stream must start on a separate line. In other words, a command can be split across more than one line using continuation marks, but the first word or string of the next command must start on a new line.

```
DO NOT DO THIS:The following input is invalid because the
               SELECT STATISTICS command (SEL STAT WHERE...)
               does not begin a new line but is continued
               from the previous line.
               SIGNON USERID=(LYNN)            -
               NODE=CD.BOSTON                  -
                      SEL STAT WHERE (PNUM=24)
               SIGNOFF

DO THIS:       The correct way to issue these commands is
               shown below.
               SIGNON USERID=(LYNN)            -
                       NODE=CD.BOSTON
               SEL STAT WHERE (PNUM=24)
               SIGNOFF
```

- The maximum command string length is 1,022 bytes for any single command. An error message appears if you exceed this length.
- The ESF feature of Connect:Direct can be specified as a SIGNON command parameter. Only SIGNON, SIGNOFF, and SUBMIT commands can be issued using ESF. If ESF is not being used, the DTF(s) that the user wishes to sign on to must be active.

Sample Job Stream to Run the Connect:Direct for VSE/ESA Batch Interface

The following is an example of the JCL required to execute the VSE/ESA Batch Interface DMBATCH.

```
*  DMBATCH OUTPUT
// DLBL DMPRINT,'SYSOUT.SYS002'
// ASSGN SYS002,SYSLST
*
*  Connect:Direct MESSAGE FILE
// DLBL DMMSGFL,'NDM.MSG',,VSAM
*
*  Connect:Direct TEMPORARY FILE
// DLBL NDMX001,'NDM.BITEMPF',0
// EXTENT SYS003,SYSWK1,1,0,600,30
// ASSGN SYS003,DISK,VOL=SYSWK1,SHR
// DLBL ESTAE,'SYSOUT.SYS011',0
// ASSGN SYS011,SYSOUT
// DLBL RPLERRC,'SYSOUT.SYS012',0
// ASSGN SYS012,SYSOUT
*
// EXEC DMBATCH,SIZE=AUTO,PARM='YYSLY'
   SIGNON USERID=(userid,password)           -
          NODE=nodename                      -
          NETMAP=NDM.NETMAP
   SUB PROC=COPY
   SIGNOFF
/*
```

The following shows a sample COPY Process that is submitted in the above sample.

```
COPY  PROCESS
STEP1 COPY FROM (PNODE DSN=&DSN1 DISP=SHR)                          -
           TO   (DSN=DATASET2  DISP=(NEW,CATLG))                    -
                COMPRESS
```

DMBATCH Parameter Descriptions

The characters shown in the PARM keyword on the DMBATCH EXEC statement are required. These parameters identify the optional output formatting routines, which are defined in the following table. These fields are positional, and there are no default values. Connect:Direct displays the output to the data set defined by DMPRINT.

Position	Value	Meaning
1	Y	Displays the command string that was executed
	N	Does not display the command string that was executed
2	Y	Displays the API return code and message ID

KEY:Y–Yes; N–No; S–Short text; L–Long text

Position	Value	Meaning
	N	Does not display the API return code and message ID
3	S	Displays short message text when there is a return code of zero from the API
	L	Displays long message text when there is a return code of zero from the API
	N	Does not display message text when there is a return code of zero from the API
4	S	Displays short message text when there is a nonzero return code from the API
	L	Displays long message text when there is a nonzero return code from the API
	N	Does not display message text when there is a return code greater than zero from the API
5	Y	Displays the data that was generated in the temporary file
	N	Does not display the data that was generated in the temporary file
6	N	Reserved
7	N	Reserved

KEY:Y–Yes; N–No; S–Short text; L–Long text

Interpreting DMBATCH Return Codes with DMCHLAPI

DMBATCH communicates with the Connect:Direct API through a high-level application interface program, DMCHLAPI. The DMCHLAPI program detects error situations while processing Connect:Direct commands for DMBATCH and issues special return codes based on these situations. If you receive a DMBATCH return code greater than **8**, check the following list for error information. The DMBATCH return codes are shown in the following table.

Return Code	Meanings
0000000C	A session was lost in a multiple-session environment.
00000014	The master session was signed off successfully.
00000018	The master session signon failed, and there is no ESF facility.
0000001C	A non-master signon failed.
00000020	The high-level interface program DMCHLAPI received an invalid number of input parameters.
00000024	The output specification included an invalid parameter.
00000028	Invalid pointer to the UICB was passed to DMCHLAPI.

Note: Always check DMPRINT output for verification of the processing situation that has just occurred.

Notifying Users of Process Status

Connect:Direct provides sample programs that notify the console operator regarding the success or failure of any Process step.

RUN TASK and MODAL statements can be used with the sample programs to notify users. For Connect:Direct for VSE/ESA, the sample programs follow:

✦ Source statement form in member DMNOTIFY.S

✦ Load module form in member DMNOTIFY.PHASE

Operator Console Notify Program (DMNOTIFY)

The Connect:Direct for VSE/ESA operator console program, DMNOTIFY, works with the RUN TASK statement to notify the console operator of the success or failure of Connect:Direct for VSE/ESA Processes by means of write-to-operator (WTO) messages. The program must have the following two parameters passed to it:

✦ GOOD or FAIL parameter telling the program the correct message to output

✦ Name of the file that is being copied

The following example demonstrates a Connect:Direct for VSE/ESA Process using the DMNOTIFY program:

```
 COPY1     PROCESS PNODE=CDA SNODE=CDB
STEP01   COPY                                  -
      FROM (                                   -
         PNODE                                 -
         DSN=FILE1                             -
         DISP=SHR                              -
         )                                     -
      TO  (                                    -
         SNODE                                 -
         DSN=FILE2                             -
         DISP=SHR                              -

      IF   (STEP01=0) THEN
         RUN TASK (PGM=DMNOTIFY,               -
            PARM=('GOOD',FILE1)) PNODE
                ELSE
                RUN TASK (PGM=DMNOTIFY,                              -
            PARM=('FAIL',FILE1)) PNODE
                EIF
```

The DMNOTIFY program is attached on the PNODE. Connect:Direct passes a parameter list containing GOOD or FAIL message criteria and the FILE1 file name to DMNOTIFY. If the COPY statement is successful, the first RUN TASK step executes. If the statement is unsuccessful, the second RUN TASK step executes.

The following messages appear for the GOOD value:

```
********  Connect:Direct FILE TRANSFER SUCCESSFUL ********
********  Connect:Direct FILE TRANSFER SUCCESSFUL ********
FILE XFER BETWEEN - CDA AND - CDB SUCCESSFUL
  REQUESTOR=userid
  DSN=FILE1
********  Connect:Direct FILE TRANSFER SUCCESSFUL ********
********  Connect:Direct FILE TRANSFER SUCCESSFUL ********
```

The following messages appear for the FAIL value:

```
********  Connect:Direct FILE TRANSFER FAILED ********
********  Connect:Direct FILE TRANSFER FAILED ********
FILE XFER BETWEEN - CDA AND - CDB FAILED
   REQUESTOR=userid
   DSN=FILE1
********  Connect:Direct FILE TRANSFER FAILED ********
********  Connect:Direct FILE TRANSFER FAILED ********
```

Performing VSAM Utility Functions

The VSAM AMS interface program, DMRTAMS, is in the installation library and can dynamically invoke AMS to perform VSAM utility functions. You can use any valid input to the VSAM AMS program as input to this program, even multiple input parameters defining one or more AMS actions. The Connect:Direct RUN TASK command invokes DMRTAMS with parameters that define allocation and AMS actions to be executed.

The first parameter is significant because it defines dynamic allocation parameters for the AMS SYSPRINT output. The following example explains how the output is routed to VSE/POWER:

```
C'MSG=YES DSN=SYSOUT.SYSxxx DD=SYSOUT DISP=SHR'
```

All VSAM control statement parameters must begin with a blank.

The parameter connects a VSE/ESA DLBL/ASSGN combination to the routing of the output of DMRTAMS to VSE/POWER. The VSE/POWER SYSLST output for the Connect:Direct DTF contains the output for the IDCAMS VSAM definition.

DD=SYSOUT defines the DLBL FILE NAME. DSN=SYSOUT.SYSxxx defines the DLBL FILE ID where SYSxxx is an unused number in the Connect:Direct partition. (The partition where the DTF is running.) Continue to use SYSOUT for the FILE NAME and FILE ID portion as shown.

A DLBL and ASSGN statement MUST be placed in the startup JCL for Connect:Direct for VSE/ESA to allow proper routing of the output to POWER. NO extent statement is needed.

Note: The highest completion code encountered in any AMS action will be returned.

Sample Startup JCL

The following example illustrates the startup JCL.

```
// DLBL SYSOUT,'SYSOUT.SYS034'
// ASSGN SYS034,SYSLST
```

Sample DMRTAMS Parameter Statement

The following example illustrates the DMRTAMS parameter statement.

```
C'MSG=YES DSN=SYSOUT.SYS034 DD=SYSOUT DISP=SHR'
```

Sample VSAM AMS Define Cluster Program

The following figure shows an example of a VSAM AMS Define Cluster.

```
VADC1    PROCESS              SNODE=BOSTON
DEFINE   RUN TASK             (PGM=DMRTAMS,                          -
              C'MSG=YES DSN=SYSOUT.SYS034 DD=SYSOUT DISP=SHR'),      -
              C' DELETE(VSAM.PAY.CHECKS)CLUSTER',                    -
              C' DEFINE CLUSTER                -',                   -
              C' (NAME(VSAM.PAY.CHECKS)        -',                   -
              C' RECORDS(1)                    -',                   -
              C' VOLUMES(DCM009)               -',                   -
              C' OWNER(NDM)                    -',                   -
              C' NONINDEXED                    -',                   -
              C' RECORDSIZE(4089 4089)         -',                   -
              C' SHAREOPTIONS (2))             -',                   -
              C' DATA                          -',                   -
              C' (CONTROLINTERVALSIZE(4096)    -',                   -
              C' NAME(VSAM.PAY.CHECKS.DATA))   -',                   -
              C' CATALOG(USER.UCAT)')) SNODE
```

Running the VSE Librarian Dynamically

The dynamic LIBR program, DMRTLIBR, in the installation library can dynamically execute the VSE Librarian program (LIBR) within a Process.

The first parameter is significant because it defines the dynamic allocation parameters for the LIBR SYSLST output. The following example shows how the output is routed to VSE/POWER:

```
C'MSG=YES DSN=SYSOUT.SYSxxx DD=SYSOUT DISP=SHR'
```

The parameter connects a VSE DLBL/ASSGN combination to the routing of the output of DMRTLIBR to VSE/POWER. The VSE/POWER SYSLST output for the Connect:Direct DTF contains the output for LIBR commands.

DD=SYSOUT defines the DLBL FILE NAME.

DSN=SYSOUT.SYSxxx defines the DLBL FILE ID where SYSxxx is an unused system number in the Connect:Direct partition where the DTF is running.) Continue to use SYSOUT for the FILE NAME and FILE ID portion as shown.

A DLBL and ASSGN statement MUST be placed in the startup JCL for Connect:Direct for VSE/ESA to allow proper routing of the output to VSE/POWER. No EXTENT statement is needed.

Completion Code

The completion code is the highest LIBR completion code from all the LIBR control statements provided.

LIBR Parameters

The RUN TASK command invokes DMRTLIBR with parameters that define the LIBR functions required. The library names referenced in the LIBR control statements must have been defined in the Connect:Direct startup job stream, or in the standard labels.

- Perform each LIBR function with as many parameters as required.
- Parameters must not extend past column 72, or they will not be read.

Sample Program Uses

The following examples demonstrate how you might use the function of the DMRTLIBR program.

Example: Delete Locked Members from a Sublibrary

The following example calls the program DMRTLIBR to unlock a specific member and to delete the member. The LIBDD DLBL/EXTENT must be in the Connect:Direct startup JCL or in the standard labels.

```
PROC1 PROC
ALLOC RUN TASK (PGM=DMRTLIBR                                         -
PARM=(C'MSG=YES DSN=SYSOUT.SYS010 DD=SYSOUT DISP=SHR',               -
      C' A S=LIBDD.SUBLIB1',                                         -
      C' UNLOCK MEMBER1.TYPE LOCKID=MYLOCK ',                        -
      C' DELETE MEMBER1.TYPE ' ) )
```

Issuing VSE/POWER Commands Dynamically

The VSE/POWER interface program, DMRTPOWR, is in the installation library and can dynamically issue VSE/POWER commands, and issue segmentation requests to VSE/POWER spooled LST or PUN data sets within the Connect:Direct partition. You can use any VSE/POWER command that is valid when issued through the VSE/POWER Spool Access Services interface. Refer to the IBM VSE/POWER documentation for additional information about this interface.

If you desire to override the number of times DMRTPOWR attempts (default=10) to connect to VSE/POWER and the amount of delay time between each attempt (default=10 seconds), then specify the first two parameters as:

```
F"nnnn", F"nnnn"
```

If your installation requires a VSE/POWER master password, specify this parameter next:

```
C"MPWD=masterpassword"
```

To pass a command to VSE/POWER, specify:

```
C"VSE/POWER command"
```

To pass a segmentation request to VSE/POWER, specify:

```
C"SYSnnn=* $$LST ..."  nnn can be "LST", "PUN" or 000-255.
```

Sample VSE/POWER RUN TASK to delete two jobs from the LST queue:

```
RTPOWER  PROCESS  SNODE=nodename
STEP01   RUN TASK                                PNODE -
                 (PGM=DMRTPOWR, PARM=(                 -
                  C"L LST,POWRJOB1",                   -
                  C"L LST,POWRJOB2"))
```

Sample VSE/POWER RUN TASK to segment Connect:Direct output to SYS010:

```
RTPOWER PROCESS   SNODE=nodename
STEP02  RUN TASK                                   PNODE -
                  (PGM=DMRTPOWR, PARM=(                  -
                   C"SYS010=* $$LST DISP=H") )
```

DMRTPOWR Return Codes

The following table lists the DMRTPOWR return codes and defines each code.

Return Code	Meaning
0000	No errors
001C	Number of delays specified incorrectly

Return Code	Meaning
0020	Second parm not fullword
0022	MPWD specified incorrectly
0028	GETVIS for SPL failed
002C	VSE/POWER command given too long
0030	Incorrect SYS number specified
xxyy	xx=XPCC return code, yy=XPCC reason code

Symbolic Resolution Utility (DMRTSUB)

The Connect:Direct utility program, DMRTSUB, allows for job submission to the internal reader, much like RUN JOB. DMRTSUB has the added capability of symbolic resolution into the job. The RUN TASK statement invokes DMRTSUB.

Parameters

Parameter	Description
JCL source	The first parameter is an allocation string to point to the job to be submitted. You should code this parameter as DSN=type(membername) or DSN=membername.
Subsequent Parameters	Subsequent parameters define resolutions for symbols used in the JCL itself. For each parameter, the first position is the name of the symbol found in the JCL and the second position is the value to be used. Separate the first and second positions by an equal sign (=), one or more spaces, or a comma. The value is terminated by the end of the parameter string, or a comma, or a comma or space not within a quoted string. A parameter may also contain only a symbol name, in which case the value to be substituted into the job stream is a null string.

Example Using DMRTSUB

The following figure is an example of RUN TASK using the DMRTSUB program.

```
RUN TASK (PGM=DMRTSUB -
          PARM=('DSN=type(member)',                          -
          'JOB     X32JOB',                                  -
          'CLASS   A',                                       -
          'FILE    X32.FILE'                                 -
           ))  PNODE
```

The jobstream source that corresponds to this RUN TASK statement displays as the following figure. The values requiring substitution are presented in bold type.

```
* $$ JOB JNM=&JOB,CLASS=&CLASS
// JOB &JOB
// DLBL FILEIN,'&FILE'
// EXEC UTILITY
/&
* $$ EOJ
```

The following figure shows the resulting job stream after substitution has taken place. Connect:Direct replaces the variables that began with **&** with the assigned values in the DMRTSUB parameter list.

```
* $$ JOB JNM=X32JOB,CLASS=A
// JOB X32JOB
// DLBL FILEIN,'X32.FILE'
// EXEC UTILITY
/&
* $$ EOJ
```

Connect:Direct Exits and DMRTSUB

If your Connect:Direct system uses any of the standard exits (Security, RUN JOB, or RUN TASK), DMRTSUB will function with any or all of them without further modification. Both a RUN TASK and a RUN JOB exit are invoked by DMRTSUB, and any processing that takes place is done when using DMRTSUB.

Error Output

DMRTSUB will write any internal error information to the data set specified on the ESTAE DLBL statement. Always include this DLBL statement in the Connect:Direct start up jobstream.

Return Codes

The DMRTSUB return codes and their meanings are listed in the following table.

Return Code	Meanings
004	A job stream record was truncated after substitution. Symbolic substitution is performed on columns 1-71 of the JCL record. Truncation is indicated when substitution causes the data in columns 1-71 to expand beyond column 71.
008	An error occurred parsing the parameters or the job stream source.
	A symbol length is greater than 19.
	A symbol name is absent in the parameter string.
012	No parameters were passed.
	An I/O error occurred.
	An input data set allocation parameter was not found.
	The symbol table overflowed.
	A file open error occurred.
	An input record length was greater than 256.
	A member locate error occurred.

Chapter 7

Using VSE/POWER Support

Through VSE/POWER support, Connect:Direct for VSE/ESA allows you to specify VSE/POWER LST and PUN queue entries as source or target files in user-written Processes. VSE/POWER LST or PUN queue entries can also trigger automatic submission of Processes. You can also automate the schedule for when members come.

Transferring to VSE/POWER

This section explains how to transfer to and from VSE/POWER.

Transferring to the RDR Queue

All of the VSE/POWER characteristics of a job are in the card images submitted to the VSE/POWER RDR queue, including class and job name. The TO DSN is a dummy name. The data set being transferred to the VSE/POWER RDR queue must contain all the necessary JECL and JCL.

Transferring to the LST or PUN Queue

All of the queue entry characteristics specified through JOB, LST or-PUN statements can be supplied by keywords within the Process using the LST= or PUN= keyword. VSE/POWER defaults are used for any parameters you do not specify.

Transferring from VSE/POWER

When transferring from a VSE/POWER queue entry, the DSN= parameter must specify the job name, the job number and optionally the segment number as illustrated in the following figure:

```
DSN=jobname.jobnumber.segnumber
```

If the job number is specified as **0** (or 00000), the first queue entry with the specified job name and class is retrieved.

LST= or PUN= must specify the class of the queue entry. LST= or PUN= may be specified on the Process if the Process is submitted from a VSE/ESA platform. If the Process is not submitted from a VSE/ESA platform, LST= or PUN= must be specified within SYSOPTS=.

Queue entries do not have to be dispatchable in order to be retrieved by the VSE/POWER support. However, if the queue entry is dispatchable (DISP=D|K), the VSE/POWER support will either delete the queue entry (DISP=D) or change the disposition to **L** (DISP=K) when the queue entry is processed, unless DISP= has been specified in the Process.

VSE/POWER LST and PUN queue entries are processed in browse mode, which allows the same queue entry to be processed at the same time by multiple Processes. Queue entries currently being accessed in other than browse mode may not be accessed by the Connect:Direct VSE/POWER support.

Automating Transfers from LST or PUN Queue

Connect:Direct for VSE/ESA uses the VSE/POWER Device Driver support for automatic transfers of queue entries as they arrive in the VSE/POWER LST and PUN queues. You can specify up to 8 destinations and up to **4** classes for both the LST and PUN queue entries.

When you specify the appropriate initialization parameters at Connect:Direct for VSE/ESA startup, VSE/POWER notifies Connect:Direct whenever output is placed in the LST or PUN queue for the specified userids.

Connect:Direct for VSE/ESA retrieves all the characteristics of the queue entry and places the queue entry in a special DISP=Y state. This state prevents the entry from being otherwise processed. A Process is then submitted to Connect:Direct. The Process to be submitted is specified in the initialization parameters and is a combination of the procedure name specified and the class (if the procedure name is less than 8 characters and the SNODE was not specified). All of the characteristics of the queue entry pass in the submitted procedure through symbolic parameters.

Example

For example, assume the following Connect:Direct for VSE/ESA initialization parameters specifications:

```
POWER.LST.CLASS=XYZ
POWER.LST.PROC=PROC1
POWER.LST.USERID1=DALLAS
```

Assume also that the Process PROC1Z is in the Process library:

```
PROC1Z  PROCESS                                              -
        &PWRDSN=          /* jobnm.jobnumb.segnumb */         -
        &CLASS=           /* class                 */         -
        &PDISP=           /* original disposition  */         -
        &JOBNM=           /* job name              */
STEP01  COPY FROM(                                    PNODE   -
                  DSN=&PWRDSN                                 -
                  LST=( CLASS=&CLASS DISP=&PDISP )            -
                 )                                            -
        CKPT=10M                                              -
        COMPRESS EXT                                          -
        TO  (                                         SNODE   -
             DSN=MVS.TEST.PDS(&JOBNM.)                        -
             DISP=RPL                                         -
             )
        EXIT
```

Connect:Direct for VSE/ESA causes VSE/POWER to notify Connect:Direct when any dispatchable LST output of class X, Y, or Z is in the LST queue destined for userid DALLAS. If you now assume that a job produces dispatchable (DISP=D or DISP=K) output for CLASS=Z, destined for userid DALLAS, Connect:Direct will be notified as soon as that output is placed in the LST queue.

Connect:Direct for VSE/ESA retrieves all the characteristics of the queue entry such as class, priority, submitter's name, forms-id, and places the queue entry into DISP=Y. The device driver module (DMGPOWER) then signs on to Connect:Direct and submits Process PROC1Z (the Process name specified was less than 8 characters). The characteristics pass with the submitted procedure as symbolic parameters. In this example, only &PWRDSN, &CLASS, &PDISP and &JOBNM are used in the procedure.

Additional sample Processes that use Connect:Direct for VSE/ESA VSE/POWER support are in the *Connect:Direct for VSE/ESA Release Notes*.

VSE/POWER Device Driver (DMGPOWER)

The Connect:Direct for VSE/ESA VSE/POWER Device Driver (DMGPOWER) starts automatically within Connect:Direct whenever at least one POWER.LST.USERIDx or POWER.PUN.USERIDx has been specified in the Connect:Direct initialization parameters. However, DMGPOWER may also be run as a subtask of VTAM or in a separate batch partition. This allows you to have more than 8 userids for the LST and PUN queues.

DMGPOWER as a Subtask of ACF/VTAM

Use the following VTAM MODIFY command to attach DMGPOWER as a subtask of ACF/VTAM.

```
F NET,SUBTASK,ID=DMGPOWER
```

DMGPOWER must reside in a library specified in the LIBDEF PHASE SEARCH= chain. The initialization parameters must reside in the LIBDEF SOURCE SEARCH= chain as member DMGPOWER.N. The initialization parameters may include the complete

Connect:Direct for VSE/ESA initialization parameters. All non-POWER parameters are ignored.

You may terminate DMGPOWER as a subtask of ACF/VTAM by using the following VTAM MODIFY command.

```
F NET,SUBTASK,ID=DMGPOWER,FUNC=DETACH
```

DMGPOWER in a Batch Partition

DMGPOWER may also be executed in a batch partition (static or dynamic). DMGPOWER must reside in a library specified in the LIBDEF PHASE SEARCH= chain. The initialization parameters must reside in the LIBDEF SOURCE SEARCH= chain as member DMGPOWER.N unless overridden by the PARM= specification on the EXEC statement. The initialization parameters may include the complete

Connect:Direct for VSE/ESA initialization parameters. All non-POWER parameters are ignored.

In the following example, the member INIT.PARMS contains the initialization parameters.

```
// EXEC DMGPOWER,PARM=`INIT.PARMS'
```

You may terminate DMGPOWER in a batch partition by canceling the partition.

Terminating DMGPOWER

DMGPOWER running under Connect:Direct for VSE/ESA will be automatically terminated whenever Connect:Direct terminates. If you desire to terminate DMGPOWER without terminating Connect:Direct, you may terminate the VSE/POWER Spool Access connections. DMGPOWER establishes two connections to VSE/POWER as shown in the following figure:

```
CD$LSTpp  for LST output (pp is the SYSLOG id of the partition)
CD$PUNpp  for PUN output (pp is the SYSLOG id of the partition)
```

For example, if DMGPOWER were running in partition C5, you could terminate DMGPOWER in C5 by issuing the following commands:

```
P DEV,CD$LSTC5
P DEV,CD$PUNC5
```

Terminating the VSE/POWER spool access connection will terminate DMGPOWER regardless of whether DMGPOWER is running under Connect:Direct for VSE/ESA, as a subtask of ACF/VTAM, or in a batch partition.

Restarting DMGPOWER

If the VSE/POWER Spool Access connection has been stopped, you can restart it by issuing the following VSE/POWER command:

- For LST output, where pp is the SYSLOG id of the partition:

```
S DEV,CD$LSTpp,SYSLSTpp
```

- For PUN output, where pp is the SYSLOG id of the partition:

```
S DEV,CD$PUNpp,SYSPUNpp
```

Note: If you specify classes in this command, they will be ignored. Specify classes in the POWER.LST.CLASS and POWER.PUN.CLASS initialization parameters.

Variables

For each process submitted by DMGPOWER, the following variables are used when the corresponding value in the VSE/POWER LST|PUN queue element is not blank or null. Detailed information about using these variables and writing Processes is provided on the Connect:Direct Processes Web site at www.sterlingcommerce.com/Documentation/processes/processhome.html.

```
&SNODE=                  SNODE NAME
&SUBNAME=                SUBMITTERS NAME (FROM)
&JOBNM=                  JOBNAME
&JOBID=                  JOBID
&JOBSEG=                 JOB SEGMENT NUMBER
&JSEP=                   JOB SEPARATOR PAGES
&CLASS=                  CLASS
&PDISP=                  DISPOSITION
&DEST=                   DEST
&DIST=                   DIST
&COPIES=                 COPIES
&FORM=                   FORM
&UCS=                    UCS
&UCS=                    UCS
&FCB=                    FCB
&BURST=                  BURST
&CMTTC=                  CMTABLE
&CMOD=                   CMOD
&FLSCT=                  FLASHC
&FLASH=                  FLASH
&FORMDEF=                FORMDEF
&PAGEDEF=                PAGEDEF
&COPYG=                  COPYG
&CHARS=                  CHARS
&PWRDSN=                 PWRDSN
&BLDG=                   BUILDING
&CKPTL=                  CKPTLINE
&CKPTP=                  CKPTPAGE
&CKPTS=                  CKPTSEC
&CNTL=                   DEFAULT SPACING
&COMPACT=                COMPACTATION TABLE
&DATCK=                  DATACK
&DEFAULT=                DEFAULT
&DEPT=                   DEPARTMENT
&LINDEX=                 LINDEX
&PROGR=                  PROGRAMMER NAME
&PRMODE=                 PRMODE
&PRTY=                   PRTY
&ROOM=                   ROOM
&THRES=                  THRESHLD
&TRC=                    TRC
&WRITER=                 WRITER
&NOTIFY=                 1ST NOTIFY ID
&USER=                   USER DATA
&CICSDATA=               CICSDATA
&QUEUE=                  LST|PUN
```

Sample Process

The following figure presents a sample Process for a VSE/POWER spool transfer. For additional examples, see the Connect:Direct Processes Web site at www.sterlingcommerce.com/Documentation/processes/processhome.html.

```
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/*
/*          C:D-VSE VSE/POWER SPOOL TRANSFER SAMPLE PROCESS
/*
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
CDPROCES PROC   SNODE=SS.DUB.DALLAS,                          -
                &SNODE=,                                      -
                &BLDG=,          /* BUILDING               */ -
                &BURST=,         /* BURST=YES OR NO        */ -
                &CHARS=,         /* CHAR ARRANGEMENT TABLE */ -
                &CICSDATA=,      /* CICS DATA              */ -
                &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)          */ -
                &DEFAULT=,       /* DEFAULT                */ -
                &DEPT=,          /* DEPARTMENT             */ -
                &DEST=,          /* OUTPUT DESTINATION     */ -
                &DIST=,          /* POWER DISTRIBUTION     */ -
                &FORM=,          /* OUTPUT FORM            */ -
                &FCB=,           /* OUTPUT WTR FCB         */ -
                &FLASH=,         /* FLASH FORMS OVERLAY    */ -
                &FLSCT=,         /* FLASH FORMS OVERLAY CNT*/ -
                &FORMDEF=        /* FORMDEF                */ -
                &HOLD=,          /* OUTPUT HOLD Y/N        */ -
                &JOBID=,         /* JOB NUMBER             */ -
                &JOBNM=,         /* JOB NAME               */ -
                &JOBSEG=,        /* SEGMENT NUMBER         */ -
                &JSEP=,          /* NUMBER OF SEP PAGES    */ -
                &LINDEX=,        /* LINDEX                 */ -
                &NOTIFY=,        /* NOTIFY ID              */ -
                &PAGEDEF=,       /* PAGEDEF                */ -
                &PDISP=,         /* POWER DISPOSITION      */ -
                &PRMODE=,        /* PRMODE                 */ -
                &PROGR=,         /* PROGRAMMER NAME        */ -
                &PRTY=,          /* PRTY                   */ -
                &PWRDSN=,        /* POWER DATASETNAME      */ -
                &QUEUE=,         /* QUEUE (LST/PUN)        */ -
                &ROOM=,          /* ROOM                   */ -
                &SUBNAME=,       /* SUBMITTERS NAME        */ -
                &THRES=,         /* THRESHLD               */ -
                &TRC=,           /* TRC                    */ -
                &UCS=,           /* UCS                    */ -
                &USER=,          /* USER INFORMATION       */ -
                &WRITER=,        /* WRITER                 */
```

(continued)

```
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/*   COPY FILE
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
STEP01   COPY FROM(                                     PNODE -
                   DSN=&PWRDSN                                -
                   LST=( CLASS=&CLASS )                       -
                   )                                          -
                   CKPT=1M                                    -
                   COMPRESS EXT                               -
              TO  (                                     SNODE -
                   DSN=DALLAS.TEST.PDS(&JOBNM.)               -
                   DISP=RPL                                   -
                   )
STEP02   IF     (STEP01 ¬= 0) THEN
         ELSE
         EXIT
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/*   SEND IT BACK TO VSE/POWER
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
STEP03   COPY FROM(                                     SNODE -
                   DSN=DALLAS.TEST.PDS(&JOBNM.)               -
                   DISP=SHR                                   -
                   )                                          -
                   CKPT=1M                                    -
                   COMPRESS EXT                               -
              TO  (                                     PNODE -
                   DSN=&JOBNM DCB=(RECFM=FM)                  -
                   LST=(CLASS=X,HOLD=YES)                     -
                   )
         EIF
STEP04   IF     (STEP03 ¬= 0) THEN
         EXIT
         ELSE
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/*   DELETE THE FILE FROM VSE/POWER
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
STEP05   RUN TASK (PGM=DMRTPOWR,                        PNODE -
                   PARM=("L LST,&JOBNM.,CCLASS=X")            -
                   )
         EIF
         EXIT
```

Chapter 8

Using VSE Librarian Support

Connect:Direct for VSE/ESA supports the VSE Librarian. You can use this feature to create multiple system administrators and to automate your daily operations. You can also copy LIBR members to VSE/ESA and non-VSE/ESA platforms.

Transferring VSE Librarian Members

Connect:Direct for VSE/ESA allows you to specify VSE Librarian members as the source or the target in a user-written Process. Connect:Direct supports all member types, including PHASE and DUMP. These two member types are in internal VSE/ESA format and are only meaningful when transferred to another VSE/ESA platform.

Identifying the VSE Library to Connect:Direct

When you use the VSE Librarian, you must identify the library to the Connect:Direct Process. The presence of the LIBR= keyword in the Connect:Direct COPY statement notifies Connect:Direct that a VSE library is to be processed. You can define parameters within the LIBR= keyword to specify:

- Selection criteria
- Exclusion criteria
- Sublibrary
- Type information

For detailed information about these parameters, see the VSE COPY statement on the Connect:Direct Processes Web site at www.sterlingcommerce.com/Documentation/processes/processhome.html.

Transfers to Another VSE/ESA Platform

When you transfer files between VSE platforms, you may transfer a single member, the entire contents of a library, or any amount between these two values. Your Process can transfer the members into an existing library/sublibrary or define a new library/sublibrary. You can also transfer librarian members to other file types.

Transfers to a Non-VSE/ESA Platform

You can also transfer VSE Librarian members to non-VSE platforms, such as MVS. You can transfer the members as sequential files or to members of a PDS. Because the VSE Librarian has more detail, or granularity, than the MVS PDS, the members of a single library, sublibrary, and type may be viewed as a PDS.

For non-VSE platforms, you specify the LIBR= keyword and its parameters within the SYSOPTS= keyword on the COPY statement.

Sample Transfers Using the VSE Librarian

The following examples show how to use the VSE Librarian in a variety of scenarios.

Transferring VSE Library Members to an MVS PDS

In the following example, all the members from the sublibrary MYSUB and of type **A** are sent to an MVS PDS. Duplicate members in the MVS PDS are not replaced.

```
TOPDS    PROCESS
STEP01   COPY FROM(DSN=VSE.LIBRARY                  PNODE -
                   DISP=SHR                               -
                   DCB=(DSORG=VSAM)                       -
                   UNIT=DISK                              -
                   LIBR=(SELSLIB=MYSUB                    -
                         SELTYPE=A                        -
                         REPLACE=NO) )                    -
              TO  (DSN=MVS.TEST.PDS                 SNODE -
                   DISP=SHR)
```

Transferring MVS PDS Members to a VSE Library

In the following example, all members of an MVS PDS are transferred to an existing VSE library and placed in a sublibrary NEWSUB. If the sublibrary NEWSUB already exists, it will be replaced. If it does not exist, NEWSUB will be created. The Process will be submitted from an MVS. Since the Process is submitted on a non-VSE platform, then you must code the LIBR= keyword and parameters must be coded within the SYSOPTS= keyword.

```
FROMPDS  PROCESS
STEP01   COPY FROM(DSN=MVS.PROCESS.LIB              PNODE -
                   DISP=SHR )                             -
              TO  (DSN=VSE.LIBRARY                  SNODE -
                   DISP=SHR                               -
                   DCB=(DSORG=VSAM)                       -
                   SYSOPTS=(LIBR=(SUBLIB=NEWSUB           -
                                  TYPE=N                  -
                                  SLIBDISP=RPL) ) )
```

Transferring a Single MVS PDS Member to a VSE Library

In the following example, a member of an MVS PDS is transferred to an existing VSE library and placed in an existing sublibrary OLDSUB. The cataloged member is to be under MSHP control. The Process will be submitted from MVS. Since the Process is submitted on a non-VSE platform, then you must code the LIBR= parameters within the SYSOPTS= keyword.

```
FROMPDS  PROCESS
STEP01   COPY FROM(DSN=MVS.PROCESS.LIB(MEMBER1)     PNODE -
                   DISP=SHR)                              -
              TO  (DSN=VSE.LIBRARY                  SNODE -
                   DISP=SHR                               -
                   DCB=(DSORG=VSAM)                       -
                   SYSOPTS=(LIBR=(SUBLIB=OLDSUB           -
                                  TYPE=N                  -
                                  MSHP=YES                -
                                  SLIBDISP=SHR) ) )
```

Transferring an Entire VSE Library to Another VSE System

In the following example, an entire VSE library will be transferred to another VSE/ESA system. The library will be created on the target VSE/ESA system. All members of all types in all sublibraries are to be transferred. The new library will be defined even if it already exists and has members in it that are locked. The members in the resulting library will have the same LOCK and same MSHP controlled information.

```
VSETOVSE PROCESS
STEP1    COPY    FROM(DSN=VSE.BAM.LIBRARY       PNODE -
                      DISP=SHR                        -
                      LIBR=(*)                        -
                      )                               -
                 TO  (DSN=VSE.NEW.BAM.LIBRARY   SNODE -
                      UNIT=DISK,                      -
                      VOL=SER=USER03,                 -
                      SPACE=(552,(22))                -
                      DISP=RPL                        -
                      LIBR=(*) )
```

For additional sample Processes that use the VSE Librarian support, refer to the *Connect:Direct for VSE/ESA Release Notes*. These sample Processes are also in your installation library and can be printed or punched.

Glossary

A

ADJACENT.NODE

Adjacent node entries in the network map define nodes in the network with which the local Connect:Direct may communicate. Each entry specifies a locally used Connect:Direct name, its associated network communications name, and session control parameters for these nodes.

API (Application Program Interface)

This Connect:Direct component accepts commands from the Interactive User Interface (IUI), Connect:Direct Batch Interface, the console operator interface, or user-written program and places the commands in a format so that the user's request can be executed by the DTF. If there are errors, the API returns a message to the user. If there are no errors, the API sends the command to the DTF using a VTAM session.

APPLID

The APPLID is the name specified in the ACB macro that identifies the application program to VTAM. For Connect:Direct, these APPLIDS correspond to a DTF node name or an interactive or batch API APPLID.

APPLID POOL

The pool identifies the APPLIDS to be used for interactive or batch communication with the DTF.

APPLID POOL Segregation

APPLID POOL segregation separates the pools of APPLIDs for use bythe individual API types (BATCH and CICS).

API System ID(s)

This ID is the system identifier (SYSIDs or SMF IDs) of the CPUs (up to 16) that will be sharing a copy of the Connect:Direct for VSE/ESA DTF utilizing the Shared DASD Facility (SDF) option of Connect:Direct.

Authorization Facility

Connect:Direct authorization facility grants access to Connect:Direct commands.

Authorization File

Connect:Direct for VSE/ESA authorization file controls access to Connect:Direct for VSE/ESA and identifies commands that can be executed by the userid. This file can also be used in conjunction with security exit interfaces with the secure point-of-entry feature.

B

Batch Interface

This Connect:Direct interface allows users to request Connect:Direct services from a batch jobstream through control statements passed to a Connect:Direct-supplied program, DMBATCH.

C

Checkpoint/Restart

For the COPY statement, checkpoint/restart eliminates the need to retransmit an entire file in the event of a transmission failure. A value on the COPY statement or in the initialization parameters, CKPT.MODE and CKPT, specifies the checkpoint interval. If a copy procedure is interrupted, Connect:Direct will restart that copy at the last checkpoint.

For the RUN TASK Statement, checkpoint/restart determines whether the RUN TASK program is to be executed again at Process restart if Connect:Direct is unable to determine whether the program has executed. RESTART in the RUN TASK statement or the RUNTASK.RESTART initialization parameter specifies whether to restart the program at the last checkpoint.

Connect:Direct Commands

These commands initiate and monitor activity within the Connect:Direct system. You can issue commands from the IUI, the operator console, a batch job or a user application program.

Connect:Direct Process

The Connect:Direct Process is a series of statements submitted through the API to initiate Connect:Direct for VSE/ESA activity, such as copying files and running jobs. The Processes can be predefined and stored in a library.

Cross Domain

Cross domain pertains to controlling resources that involve more than one domain. A domain consists of one SSCP (System Services Control Point) and all the SNA resources under its control.

This term applies when you perform a multiple-session signon to another Connect:Direct DTF and when you establish a connection to another Connect:Direct DTF to copy files.

D

DTF (Data Transmission Facility)

The DTF is the nucleus component of Connect:Direct. The DTF controls information distribution to other Connect:Direct nodes in the network. Start-up parameters that govern the overall activity of the DTF are defined within the initialization parameters.

E

ESF (Extended Submit Facility)

The ESF allows users to queue data transfer requests to a Connect:Direct node that is not active. This allows users to submit work to Connect:Direct, even when the Connect:Direct DTF is down.

I

IUI (Interactive User Interface)

The IUI is a screen and dialog component that allows users to define and submit Connect:Direct Processes and issue Connect:Direct commands that monitor and control administrative and operations activity. An IUI is available for a CICS environment with the CICS interface provided with the Connect:Direct for VSE/ESA product.

L

LOCAL.NODE

The local node entry defines the logical Connect:Direct name of the local Connect:Direct DTF and its associated communications name. The local node entry also contains the name of the transmission queue and the SUPERUSR ID password, if specified. The local node entry is in the network map.

M

Modal Statements

Connect:Direct modal statements (IF THEN, EIF, ELSE, EXIT, and GOTO) allow you to alter the sequence of Connect:Direct Process execution based on the completion of a previous Process step.

N

Network Map

VSAM file identifying all valid Connect:Direct nodes and applids in the network. Each Connect:Direct for VSE/ESA node has a network map. There is one entry within that network map for each of the other Connect:Direct nodes to which the local Connect:Direct for VSE/ESA node can initiate a session. The entries also contain the rules or protocol the nodes use when communicating.

Node

Any site within a network from which information distribution is initiated.

O

Online Messages

Completion and error messages that are displayed online.

Operator Interface

Allows Connect:Direct commands to be issued from the VSE operator console. This interface also allows tailoring of Connect:Direct commands through a command list (CLIST) facility.

P

Parallel Sessions

Parallel sessions allow you to have two or more concurrently active sessions between the same set of two LUs. With parallel session support, Connect:Direct allows multiple, concurrent file transfers between two Connect:Direct nodes.

PNODE (Primary Node)

The PNODE is the Connect:Direct node on which the Process is being submitted. The primary node may also be referred to as the controlling or source node, but should not necessarily be interpreted as the sending node since PNODE can be the receiver. In every Process, there is one PNODE and one SNODE specified. The submitter of a Process is always the PNODE. The PNODE name can be expressed in 1-16 alphanumeric and national characters (@ # $), with imbedded periods. The first character must be alphabetic.

Process

A Process is a series of statements (which can be predefined and stored in a library) submitted through the API to initiate Connect:Direct for VSE/ESA activity, such as copying files and running jobs.

Process Statements

Process statements are used to build a Connect:Direct Process. They contain instructions for transferring files, running operating system jobs, executing programs, or submitting other Connect:Direct Processes. Process statements include COPY, RUN JOB, RUN TASK, SUBMIT, SYMBOL, and Modals (conditional logic).

R

Retry Interval

This initialization parameter specifies the interval, in minutes, that the retries mentioned in the MAX RETRIES parameter will be performed.

RUN JOB Statement

This statement, executed as a Connect:Direct Process statement, allows a JCL job to be submitted to the POWER reader queue.

RUN TASK Statement

The RUN TASK statement allows an external program to be executed under the control of Connect:Direct

S

SNA (Systems Network Architecture)

SNA is a network architecture designed to provide compatibility among a wide variety of hardware and software products so that they can be used to build complex networks. It defines protocols, standards, and message formats to which different hardware and software products must conform.

SNODE (Secondary Node)

The SNODE is a Connect:Direct node that interacts with the primary node (PNODE) during process execution. The secondary node (SNODE) may also be referred to as the participating, target or destination node. In every Process, there is one PNODE and one SNODE.

Statistics File

This file is a VSAM relative record data set that holds Connect:Direct statistics records to document the history of a Process. This file is a wrap-around file.

Statistics Facility

The Connect:Direct statistics facility records all Connect:Direct activities

SUPERUSR Password

This password is reserved for the SUPERUSR userid. The password can be 1--8 characters. Also see SUPERUSR ID.

SUPERUSR ID

Special ID that is not required to pass through normal security checking when signing on to Connect:Direct. This ID allows you to perform Connect:Direct administrative functions.

SYMBOL Statement

This Connect:Direct Process statement allows you to build symbolic substitution values.

Symbolics

Connect:Direct Processes support symbolic parameters. The parameters allow one predefined Process to be used for multiple applications. For example, the file names for a COPY could be passed to the Process by the user submitting the Process.

T

TCQ (Transmission Control Queue)

The TCQ is a VSAM relative record data set (RRDS) used to hold all Processes that have been submitted to Connect:Direct for VSE/ESA. Itcontains four logical queues:

- HOLD
- WAIT
- TIMER
- EXECUTION

TCP/IP (Transmission Control Protocol/Internet Protocol)

TCP/IP is a set of network standards that specify the details of how computers communicate, as well as a set of conventions for interconnecting networks and routing traffic.

Index