Sterling Connect:Direct for UNIX

User Guide

Version 4.1

Sterling Connect:Direct for UNIX

User Guide

Version 4.1

Note
Before using this information and the product it supports, read the information in "Notices" on page 83.

This edition applies to version 4.1 of IBM Sterling Connect:Direct for UNIX and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. Controlling and Monitoring Processes

Overview of the Command Line Interface

The Command Line Interface (CLI) enables you to submit Sterling Connect:Direct® Processes and commands from a native command line environment. You can also use the Sterling Connect:Direct Browser User Interface to perform some of these tasks.

Starting the CLI

Procedure

1. If you have not defined the NDMAPICFG environment variable, type the following command for the appropriate shell, where *d_dir* is the path to the Sterling Connect:Direct subdirectory.

   In the C shell:

   ```
   % setenv NDMAPICFG d_dir/ndm/cfg/cliapi/ndmapi.cfg
   ```

   In the Bourne or Korn shell:

   ```
   $ NDMAPICFG=d_dir/ndm/cfg/cliapi/ndmapi.cfg
   $ export NDMAPICFG
   ```

2. Type the following command to invoke Sterling Connect:Direct CLI. Type options as required:

   ```
   $ direct [-P string -s -t n -e nn -n name -p nnnnn -x -r -h -z]
   ```

Stopping the CLI

Procedure

Stop the CLI operation by typing **Control-D** or **quit** at the prompt.

CLI Commands

Refer to the following table for a description of the command options and sample command entries:

Option	Description	Value	Sample Command Entry

-P	Identifies the custom string to use at the command line prompt.
	
	If the prompt string includes spaces or special characters, enclose it in single or double quotation marks.
	
	The prompt string can also be specified in the ndmapi.cfg file. If a prompt string is specified on the command line and in the ndmapi.cfg file, **-P** takes precedence.
	
	When the default prompt (“Direct”) is overridden, the new prompt string is shown at the command line prompt and in the welcome banner display.
		text string
		
		Up to 32 characters.
			$ direct -PNewPrompt
			
			$ direct -P”Test CD on Medea”

-s	Suppresses standard output. Use this option to view only the completion status of a command.	none	$ direct -s
-t n	Enables the CLI/API trace option. The level number, **n**, identifies the level of detail in the trace output.	1 | 2 | 4 Specify one of the following level numbers: 1—Provides function entry and function exit. This is the default. 2—Provides function entry and exits and basic diagnostic information, such as displaying values of internal data structures at key points in the execution flow. 4—Enables a full trace. All diagnostic information is displayed.	$ direct -t 4

Option	Description	Value	Sample Command Entry
-e nn	Defines the error level above which the CLI automatically exits. If the returned error code is greater than the error level specified, the CLI automatically exits. Use this command within shell scripts. This parameter prevents unwanted execution of commands following a command that generates an error above the specified level. When the CLI terminates, it returns a UNIX exit code that can be tested by the shell.	0 | 4 | 8 | 16 Valid values in the error level code are: 0—Indicates successful completion. 4—Indicates warning. 8—Indicates error. 16—Indicates catastrophic error.	$ direct -e 16
-n name	Identifies the host name of the computer where the Sterling Connect:Direct server (PMGR) is running. **Note:** Invoking direct with **-p** or **-n** overrides the settings in the ndmapi.cfg file.	Sterling Connect:Direct host name	$ direct -n hostname
-p nnnnn	Identifies the communications port number for the Sterling Connect:Direct node. **Note:** Invoking direct with **-p** or **-n** overrides the settings in the ndmapi.cfg file.	1024–65535. The format is nnnnn.	$ direct -p 2222
-x	Displays command input on standard out. Use this command when debugging scripts.	none	$ direct -x
-r	Makes the Process number available to user-written shell scripts. The CLI displays a special string, _CDPNUM_ followed by a space, followed by the Process number.	none	$direct -r | grep "_CDPNUM_"
-h	Displays command usage information if a Sterling Connect:Direct command is typed incorrectly.	none	$ direct -h
-z	Appends a newline character after a prompt.	none	$ direct -z

CLI Job Control

Sterling Connect:Direct enables you to switch the CLI Process between the foreground and the background in shells that support job control. This capability

enables you to edit the text of saved Processes, issue UNIX commands, and resolve Process errors without exiting and reentering the CLI. Use the following commands to switch the CLI Process:

- Press the suspend character (**Control-Z**) to stop or suspend the CLI Process.
- Issue the **fg** command to move the CLI Process to the foreground.

Note: If you experience problems with job control, contact your system administrator for suggestions on additional UNIX commands to use.

CLI History Commands

Sterling Connect:Direct enables you to use the history commands available with UNIX. History commands do not need the semicolon (;) at the end of the command. The following table lists the available history commands:

Command	Description
!!	Repeat the last command one time.
!#n	Set the number of commands to store in the history buffer. The default history buffer size is 50 commands.
!n	Repeat command number <n> in the history buffer.
!<string>	Repeat command beginning with the string <string>.
!?	List the contents of the history buffer.

Overview of Sterling Connect:Direct Commands

You control and monitor Sterling Connect:Direct Processes using the following commands:

Note: The CMGR currently limits the size of a Process file to 60K bytes.

Command	Abbreviation	Description
submit	sub	Makes Processes available for execution.
change process	cha pro	Changes the status and modifies specific characteristics, of a nonexecuting Process in the TCQ.
delete process	del pro	Removes a nonexecuting Process from the TCQ.
flush process	flush pro	Removes an executing Process from the TCQ.
stop	stop	Stops Sterling Connect:Direct for UNIX and returns control to the operating system.
select process	sel pro	Monitors both executing Processes and Processes waiting for execution. You can specify the search criteria and the form in which the information is presented.
select statistics	sel stat	Retrieves information from the statistics file. You can specify the search criteria and the form in which the information is presented.
view process	view pro	View a Process in the TCQ where the local node is the Pnode. View process can only display Processes running on the local node since only the Pnode has the information required to display a Process.

Abbreviations for Common Sterling Connect:Direct Commands

The following table lists valid abbreviations for commonly used parameters for Sterling Connect:Direct commands:

Parameter	Abbreviation
detail	det
quit	q
recids	rec
release	rel
pname	pnam, pna
pnumber	pnum
sunday	sun
monday	mon
tuesday	tue
wednesday	wed
thursday	thu
friday	fri
saturday	sat
today	tod
tomorrow	tom

Restricting the Scripts and UNIX Commands Users Can Execute

System administrators and other network operations staff can restrict the scripts and UNIX commands that you can execute with the run task and run job Process statements. System administrators and other network operations staff can enforce the following limits on the capabilities you have with Sterling Connect:Direct:

- The capability to send or receive files; you may be limited either to sending files only or to receiving files only.
- The locations to or from which you can send or receive files; you may be limited to specific local or remote nodes.

Check with the system administrator for a list of specific restrictions for your user ID.

Sterling Connect:Direct Command Syntax

Use the same command syntax for commands typed at the CLI prompt or used as the command text parameter for an **ndmapi_sendcmd()** function. Refer to "User Exit Programs" on page 75, for details on function calls. The following conventions are used when typing commands:

- When selecting a password or user ID, do not use Sterling Connect:Direct keywords.
- Be aware that user names and file names are case sensitive.
- Type an individual command keyword in uppercase, lowercase, or mixed-case characters.
- Terminate all commands with a semicolon (;).

- When typing commands, type the entire command name or type the first three characters or abbreviate specific parameters. Refer to “Abbreviations for Common Sterling Connect:Direct Commands” on page 5for a list of abbreviations.
- Do not abbreviate Process statements and parameters.
- File names, group names, user IDs, and passwords are variable length strings and can be any length.
- A Sterling Connect:Direct node name is 1–16 characters long. The name of a record in the netmap describing a remote node is typically the remote Sterling Connect:Direct node name, but can be any string 1–256 characters long. You can also specify a remote node name as an IP address or hostname and a port number or port name.

“Generic” Parameter Value

When the word generic is specified as a parameter value in a syntax definition, provide a string that can include the asterisk (*) and question mark (?) characters. These characters provide a pattern matching or wildcard facility for parameter values. The asterisk matches zero or more characters, and the question mark matches any single character. The following sample illustrates the use of the asterisk and question mark characters:

```
PNAME = A?PROD5*
```

The generic Process name specified in the previous sample shows a specification that matches all Processes beginning with the letter A, followed by any single character in position two with the string PROD5 in positions three through seven. The asterisk takes the place of zero or more characters beginning in position eight.

“List” Parameter Value

When (list) is a parameter value, you can specify multiple parameter values by enclosing the group in parentheses and separating each value with a comma. A list can also include generic values. The following command illustrates a list:

```
(pnumber1, pnumber2, pnumber3)
```

Submitting a Process

The submit command makes Processes available for execution and enables the software to interpret the Process statements contained in the specified files.

Parameters specified in the submit command override the same parameters specified on the Process statement. There are no required parameters. However, if you do not specify a file name for the file parameter, the text of the Sterling Connect:Direct Process must follow the submit command. Following are the parameters for the submit command:

Parameter	Description	Values
file	The name of the Process file. The file name can include a path name indicating the location of the Process. This parameter must be the first parameter.	file name including the path name

Parameter	Description	Values
class	The node-to-node session on which a Process can execute. A Process can execute on the class specified or any higher session class. The default class is specified as the **sess.default** parameter of the local.node record in the initialization parameters file.	1 | n A numeric value from 1 to the value of maximum concurrent local node connections (sess.pnode.max). The default value is **1**. The value cannot be greater than the maximum number of local sessions with primary control.
crc	Determines if crc checking is performed. This parameter overrides settings in the initialization parameter, the network map, and the Process. **Note:** The user must be assigned authority to change the crc settings in the user authority file.	on | off on—Turns on crc checking. off—Turns off crc checking. The default is **off**.
hold	Determines if the Process is placed in the Hold queue. When a Process is submitted with retain=yes or retain=call, Sterling Connect:Direct ignores the **hold** parameter.	yes | no | call yes—Specifies the Process is placed in the Hold queue in HI status until it is released by a change process command. A Process submitted with hold=yes is placed on the Hold queue even if you specify a start time. no—Specifies that the Process executes as soon as resources are available. This is the default. call—Specifies that the Process is held until a connection is established between the remote node and the local node. At that time, the Process is released for execution.
maxdelay	How long the submit command waits for the submitted Process to complete execution. This parameter is useful when the command is issued by a shell script. When this parameter is specified, the script waits until the Process completes before it continues execution. The return code of the Process is stored in the $? variable if you are using the Bourne or Korn shell and in $status variable if you are using the C shell, which the shell script can use to test the results of Process execution. If you do not specify maxdelay, no delay occurs. If the time interval expires, the submit command returns a warning status code and message ID to the issuing Process or CLI/API. The Process is not affected by the time interval expiration and executes normally.	unlimited | *hh:mm:ss* | 0 unlimited—Waits until the Process completes execution. hh:mm:ss—Waits for an interval no longer than the specified hours, minutes, and seconds. 0—Waits until the Process completes execution. If you specify maxdelay=0, you get the same results as when you specify maxdelay=unlimited.
newname	A new Process name that overrides the name in the submitted Process.	A name up to 256 characters long

Parameter	Description	Values
notify	The user e-mail to receive Process completion messages. This parameter uses the rmail utility available in the UNIX System V mail facility to deliver the completion messages. **Note:** Sterling Connect:Direct does not validate the e-mail address or user ID supplied to the notify parameter. Invalid e-mail addresses and failed E-mail attempts are handled according to the local mail facilities configuration.	*username@hostname* or *user@localhost*
pacct	A string containing information about the PNODE. Enclose the string in double quotation marks.	"*pnode accounting data*" up to 256 characters
pnodeid	Security user IDs and passwords at the PNODE. The pnodeid subparameters can contain 1–64 alphanumeric characters.	*id* [, *pswd*] id—Specifies a user ID on the PNODE. pswd—Specifies a user password on the PNODE.⌂If you specify pnodeid, you must also specify id. Identify the ID first and the pswd last.
prty	The priority of the Process in the Transmission Control Queue (TCQ). A Process with a higher priority is selected for execution before a Process with a lower priority. The prty value does not affect the priority during transmission.	1–15, where fifteen is the highest priority. The default is **10**.
retain	Determines if Sterling Connect:Direct retains a copy of the Process in the TCQ. Sterling Connect:Direct assigns a Process number to the Process when it is placed in the retain queue. When the Process is run, the Process number assigned to the retain Process is incremented by one. For example, if the Process is assigned the Process number of 1445 in the retain queue, the Process number is 1446 when the Process is executed. If you specify a start time and set **retain=yes**, the Process remains in the Timer queue in HR status and is submitted at the appropriate interval. For example, when **startt=(Monday,2:00)**, the Process runs each Monday at 2:00 AM. When **startt=(,1:00)**, the Process runs daily at 1:00 AM. Sterling Connect:Direct does not provide a way to run a Process hourly. To do this, you must use the UNIX cron utility. If no start time is identified, you must issue a **change process** command to release the Process for execution. ⌂Do not code the **startt** parameter when you specify **retain=initial**.	yes | no | initial yes—Specifies that the system retains the Process in the Hold queue in HR status after execution. no—Specifies that the system deletes the Process from the TCQ after execution. This is the default. initial—Specifies that the system retains the Process in the Hold queue in HR status for automatic execution every time the Process Manager initializes.
sacct	Specifies accounting data for the SNODE. Setting this value in the submit statement overrides any accounting data specified in Process.	"*snode accounting data*" up to 256 characters. Enclose the string in double quotation marks.

Parameter	Description	Values
snode	Identifies the name of the secondary node. Setting this value overrides the snode value in the Process statement. The **snode** parameter is required either on the submit command or Process statement.	*name* | *host name* | *nnn.nnn.nnn.nnn* or nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn[*;port name* | *nnnnn*]] name—Specifies the node name of the remote node. The secondary node name corresponds to an entry in the network map file. host name—Specifies the name of the host computer where the remote Sterling Connect:Direct node is running. nnn.nnn.nnn.nnn or nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn —Specifies the IP address of the remote node in IPv4 or IPv6 format: nnn.nnn.nnn.nnn (IPv4) or nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn (IPv6). [;port number |nnnnn]—Identifies the communications port. You can only use this parameter with the host name or IP address parameters. The nnnnn value is a decimal number from 1,024–65,535.

Parameter	Description	Values
snodeid	Specifies security user IDs and security passwords on the SNODE. The snodeid subparameters can contain one or more alphanumeric characters. If Sterling Connect:Direct finds that a Process has no **snodeid** parameter or defines a **snodeid** parameter and the initialization parameter **proxy.attempt** is set to **y**, then any password specified on the **snodeid** parameter is ignored. A proxy user record is a remote user record in the userfile.cfg, which corresponds to the user name specified on the snodeid parameter. If no proxy user record exists, the **snodeid** parameter must contain a valid user name and password for a UNIX user who has a corresponding local user record in the userfile.cfg file. When **proxy.attempt=n** and no snodeid is defined, Sterling Connect:Direct uses the submitting ID and node to find a Remote User Information record in the User Authorization Information file. If Sterling Connect:Direct cannot find a match, then that user cannot send or receive files. If the initialization parameters file parameter **proxy.attempt** is set to **y**, users are not required to specify a password for the **snodeid** parameter. This capability enables the id subparameter to contain a dummy user ID to be used for translation to a local user ID on the remote system. The use of a dummy user ID offers improved security because neither the sender nor the receiver are required to use an actual user ID. Reserved keywords cannot be used in the **snodeid** field.	*id* [,*pswd* [,*newpswd*]] id—Specifies a user ID on the SNODE. pswd—Specifies a user password on the SNODE. If you specify id, you do not have to specify pswd. This capability enables the id parameter to contain a dummy ID to be used for translation to a local ID on the remote system. newpswd—Specifies a new password value. On certain platforms, the user password changes to the new value on the SNODE if the user ID and old password are correct (refer to documentation on the specific platform). If the SNODE is a UNIX node, the password does not change. If you specify pswd, you must also specify id. If you specify newpswd, you must also specify pswd. Type the values in the order of id, pswd, and newpswd.

Parameter	Description	Values
startt	Identifies the date, day, and time to start the Process. Sterling Connect:Direct places the Process in the Timer queue in WS (Waiting for Start Time) status. The date, day, and time are positional parameters. If you do not specify date or day, a comma must precede time. Do not code the **startt** parameter when you **specify retain=initial**.	[*date* | *day*] [,*hh:mm:ss* [am | pm]] date—Specifies the day (dd), month (mm), and year (yy), which you can code as mm/dd/yyyy or mm-dd-yyyy. If you only specify date, the time defaults to 00:00:00, which indicates midnight. The current date is the default. day—Specifies the day of the week. Values are today, tomorrow, yesterday, monday, tuesday, wednesday, thursday, friday, saturday, and sunday. hh:mm:ss [am | pm]—Specifies the time of day in hours (hh), minutes (mm), and seconds (ss). You can specify the hour in either 12- or 24-hour format. If you use 12-hour format, then you must specify am or pm. The default is the 24-hour format. The default value is 00:00:00, which indicates midnight. ⌂If you specify only the day value, the time defaults to 00:00:00. This means that if you submit a Process on Monday, with monday as the only startt parameter, the Process does not run until the following Monday at midnight.
&symbolic name 1 &symbolic name 2 &symbolic name n	Specifies a symbolic parameter assigned a value. The value is substituted within the Process when the symbolic parameter is encountered. The value for the symbolic parameter must be in double quotation marks if it is a keyword or contains special characters. If you want to reserve the double quotation marks when the symbolic name is resolved in the Process, enclose the double-quoted string in single quotes, for example: &filename = "'filename with spaces'" The symbolic name itself must not be a subset of any other symbolic name. (You cannot have, for example, a symbolic name called &param and another symbolic name called &parameter in the same Process.)	variable string 1 variable string 2 variable string n The symbolic name cannot exceed 32 characters. ⌂⌂

Parameter	Description	Values
tracel	Specifies the level of trace to perform for a Process. Tracing by Process can be turned on in the submit command or as part of the Process definition. If you identify the snode or pnode immediately after the trace level definition, the trace level is turned on for all Processes submitted to and from the node identified.	level = 0 |1 | 2 | 4 snode | pnode file=name level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—The basic level that provides function entry and function exit.⌂2—includes level 1 plus function arguments.⌂4—Enables a full trace. Basic diagnostic information, such as values of internal data structures at key points in the execution flow, are displayed. snode—Specifies to trace only the SNODE SMGR. pnode—Specifies to trace only the PNODE SMGR. file—Specifies the name of a file where the trace output is directed. If you do not specify a file name, the file is created in the Sterling Connect:Direct working directory with the file name CMGR.TRC. The length of the name value is unlimited.

Example - Submit a Process That Runs Every Week

The following command submits the Process named payroll:

```
submit file=payroll retain=yes startt=monday pacct="1959,dept-27";
```

Because **retain=yes** is specified in this sample, the Process is retained in the TCQ after execution. The Process starts next Monday at 00:00:00 and runs every Monday thereafter. Process accounting data is specified for the PNODE.

Example - Submit a Process with a Start Time Specified

The following command submits the Process named copyfil:

```
submit file=copyfil snode=vmcent startt=(01/01/2008, 11:45:00 am);
```

Because **startt** is specified, the Process executes on the first day of January 2008 at 11:45 a.m.

Example - Submit a Process with No File Value

The following command submits a Process without a **file** parameter value, but with the Process statements typed at the CLI command prompt:

```
Direct> sub do_copy process  snode=node1
step01 copy from (
                        file=data.data
                        pnode
                    )
            to   (
                        file=b
                        snode
                    )
      pend ;
Process Submitted, Process Number = 5
```

Example - Submit a Process and Turn On Tracing

The following command submits the Process named copy.cdp:

```
submit file=copy.cdp tracel=4 pnode;
```

Because **tracel** is specified and the **pnode** parameter is included, an SMGR and COMM full trace is performed on the Process. Trace information is written to the default file SMGR.TRC.

Changing Process Parameters

The **change process** command modifies specified parameters for a nonexecuting Process.

You specify the Processes to be changed by Process name, Process number, secondary node name, and submitter.

You can change the class, destination node, and priority. You can place a Process on the Hold queue or release a Process from the Hold queue by issuing a change process command with either the **release** or **hold=no** parameter.

If you submit a Process with a **startt** parameter, Sterling Connect:Direct places the Process on the Timer queue. If a Process fails, you can move it to the Hold queue by specifying the change process command with hold=yes. Sterling Connect:Direct then places the Process in the Hold queue in HO status. You can release the Process for execution at a later time.

You can set tracing for an existing Process by setting the **tracel** parameter to 1, 2, or 4. You can turn off tracing for a Process by setting trace1 to 0.

Specify at least one of the following search criteria parameters:

Parameter	Description	Value
pname	Locate the Process to be changed by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and Sterling Connect:Direct for z/OS®.	*name* | *generic* | (*list*) name—Specifies the Process name, up to 8 alphanumeric characters. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Locate the Process to be changed by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted.	*number from 1–99,999* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
snode	Locate the Process to be changed by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.
submitter	Locate the Processes to be changed by the node specification (the Sterling Connect:Direct node name) and user ID of the Process owner. The character length of this parameter is unlimited.	(*node specification, userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification (the Sterling Connect:Direct node name) and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.

The optional parameters for the **change process** command are the following:

Parameter	Description	Value
class	Changes the node-to-node session on which a Process can execute. A Process can execute on the class specified or any higher session class. The default class is specified as the **sess.default** parameter of the local.node record in the initialization parameters file.	The default is **1**.

Parameter	Description	Value
hold	Moves the Process to the Hold or Wait queue.	yes | no | call yes—Places the Process in the Hold queue in HO status until it is released by another **change process** command. no—Places the Process in the Wait queue in WC (Waiting for Connection) status; the Process executes as soon as resources are available. This is the default. call—Places the Process in the Hold queue in HC (Hold for Call) status until the remote node (SNODE) connects to the local node (PNODE) or another Process is submitted. At that time, Sterling Connect:Direct releases the Process for execution
newsnode	Specifies a new remote node name to assign to the Process.	new remote node specification
prty	Changes the priority of the Process on the TCQ. Sterling Connect:Direct uses the **prty** parameter for Process selection. A Process with a higher priority is selected for execution before a Process with a lower priority. The **prty** value does not affect the priority during transmission.	1–15, where 15 is the highest priority. If you do not specify **prty**, the default is **10**.
release	Releases the Process from a held state. This parameter is equivalent to **hold=no**.	none
tracel	Changes the level of trace to perform for a Process. If you identify the SNODE or PNODE immediately after the trace level definition, the trace level is turned on for all Processes submitted to and from the node identified.	level = 0 | 1 | 2 | 4 level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—Is the basic level that provides function entry and function exit.⌂2 —Includes level 1 plus function arguments.⌂4—Enables a full trace. Basic diagnostic information, such as values of internal data structures at key points in the execution flow, are displayed.

The following command changes the remote node name for the Process named cdproc to a new remote node, paris:

```
change process pname=cdproc newsnode=paris;
```

Deleting a Process from the TCQ

The **delete** process command removes a nonexecuting Process from the TCQ.

You select the Process to **delete** by Process name, Process number, secondary node name, submitter, or any combination of the search criteria parameters. Specify at least one of the following search criteria parameters:

Parameter	Description	Value
pname	Identify the Process to delete by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and for z/OS.	*name* | *generic* | (*list*) name—Specifies the Process name up to 8 alphanumeric characters long. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Identify the Process to delete by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted. Valid Process numbers range from 1–99,999.	*number* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma (,).
snode	Identify the Process to delete by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.
submitter	Identify Processes to delete by the node specification and user ID of the Process owner. The character length of this parameter is unlimited.	(*node specification, userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.

The following command deletes all nonexecuting Processes submitted by user ID cduser on node dallas:

```
delete process submitter=(dallas, cduser);
```

Removing a Process from the Execution Queue

The **flush process** command removes Processes from the Execution queue. You select the Process to remove by Process name, Process number, secondary node name, submitter, or any combination of the search criteria parameters. Specify at least one of the following search criteria parameters:

Parameter	Description	Value
pname	Locate the Process to remove by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and Sterling Connect:Direct for z/OS.	*name* | *generic* | (*list*) name—Specifies the Process name, up to 8 alphanumeric characters. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Locate the Process to remove by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted.	*number from 1–99,999* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma.
snode	Locate the Process to remove by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
submitter	Locate the Processes to remove by the node specification (the Sterling Connect:Direct node name) and user ID of the Process owner.	(*node specification, userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification (the Sterling Connect:Direct node name) and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.

The flush process command has the following optional parameters:

Parameter	Description	Value
force	Forcibly terminates an executing Process or terminates a Process in an orderly fashion as the step completes. This parameter is useful if a Process is in the executing state and waiting for unavailable resources.	yes | no yes—Specifies to forcibly and immediately terminate the Process. The SMGR also terminates immediately. no—Specifies to terminate the Process in an orderly fashion as the step completes. The SMGR closes the statistics file and then terminates. This is the default.
hold	Places the terminated Process in the Hold queue where it can be released for re-execution.	yes | no yes—Specifies to place the Process in the Hold queue in HS status after the Process is terminated. no—Specifies to delete the Process from the TCQ after the Process is terminated. This is the default.

The following command flushes all executing Processes named “Rome” from the Execution queue:

```
flush process pname=rome force=yes;
```

The following command flushes all executing Processes on node alma submitted by user ID jones:

```
flush process submitter=(alma, jones);
```

Stopping Sterling Connect:Direct

The **stop** command initiates an orderly Sterling Connect:Direct shutdown sequence or forcibly terminates the software. After you run the stop command, no new Processes are allowed to run and no new connections with remote systems are established. Commands can be issued and users can sign on until the server terminates.

You can specify the force, immediate, quiesce, or step parameters with the stop command.

Note: The force parameter is required when running Sterling Connect:Direct with the LU6.2 feature on any supported platform other than AIX.

Following are the parameters for the stop command:

Parameter	Description
force	Forcibly terminates Sterling Connect:Direct and returns control to the operating system.
immediate	Begins an immediate, but orderly shutdown of all activity and terminates Sterling Connect:Direct. The software terminates connections, writes statistics records, closes files, and shuts down.
quiesce	Runs all executing Processes to completion before shutting down Sterling Connect:Direct. No new Processes are started. This is the default value.
step	Shuts down Sterling Connect:Direct after all currently executing Process steps are complete. The software writes statistics records, closes files, and shuts down. All active Processes are retained in the TCQ. Processes restart when the software is re-initialized.

The following command forcibly terminates Sterling Connect:Direct and returns control to the operating system:

```
stop force;
```

Viewing a Process in the TCQ

The **view process** command is used to view Processes in the TCQ when the local node is the PNODE. You can search by Process name, Process number, queue, secondary node, status, owner of the Process, or any combination of the search criteria parameters.

You also can specify more than one Process in the search criteria.

There are no required parameters for this command. If you do not specify an optional parameter, Sterling Connect:Direct selects all Processes executing or waiting for execution. Following are the optional parameters for the view process command:

Parameter	Description	Value
pname	Locate the Process to view by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and Sterling Connect:Direct for z/OS.	*name* | *generic* | (*list*) name—Specifies the Process name, up to 8 alphanumeric characters. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Locate the Process to view by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted.	*number from 1–99,999* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma.
queue	Specifies the Processes to be viewed by the specified queue names.	all | exec | hold | wait | timer all—Selects Processes from all queues. This is the default. exec—Selects Processes from the Execution queue. hold—Selects Processes from the Hold queue. timer—Selects Processes from the Timer queue. wait—Selects Processes from the Wait queue.
snode	View the Process by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
status	Specifies the Processes to be viewed by Process status. If you do not specify a status value, information is generated for all status values.	EX | HC | HE | HI | HO | HR | HS | PE | WC | WR | WS | (list) EX (Execution)—Specifies to select Processes from the Execution queue. HC (Held for Call)—Specifies to select Processes submitted with hold=call. HE (Held due to Error)—Specifies to select Processes held due to a connection error. HI (Held Initially)—Specifies to select Processes submitted with hold=yes. HO (Held by Operator)—Specifies to select Processes held by a **change process** command issued with hold=yes. HR (Held Retain)—Specifies to select Processes submitted with retain=yes or retain=initial. HS (Held Due to Execution Suspension)—Specifies to select Processes suspended by a **flush process** command issued with hold=yes. PE (Pending Execution)—Specifies to select Processes submitted with a **maxdelay** parameter and assigned PE status by the Process Manager just before a Session Manager is created to execute the Process. After the Session Manager initializes, the Process is placed on the Execution queue and the status is changed to EX. WC (Waiting for Connection)—Specifies to select Processes that are ready for execution, but that all available connections to the remote node are in use. WR (Waiting for Restart)—Specifies to select Processes that are waiting for restart after session failure. WS (Waiting for Start Time)—Specifies to select Processes waiting for a start time. These Processes are on the Timer Queue. list—Specifies a list of status values. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
submitter	Locate the Processes to view by the node specification (the Sterling Connect:Direct node name) and user ID of the Process owner. The length of this parameter is unlimited.	(*node specification, userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification (the Sterling Connect:Direct node name) and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.

The following command displays the specified Process number:

```
view process pnumber=1;
```

Monitoring Process Status in the TCQ

The select process command displays information about Processes in the TCQ.

The search criteria provide flexibility in selecting Processes. You can search for a Process by Process name, Process number, queue, secondary node, status, owner of the Process, or any combination of the search criteria parameters.

You also can specify more than one Process in the search criteria. You can request either a detailed report about the selected Process or a short report.

There are no required parameters for this command. If you do not specify an optional parameter, Sterling Connect:Direct selects all Processes executing or waiting for execution. Following are the optional parameters for the **select process** command:

Parameter	Description	Value
pname	Locate the Process to select by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and Sterling Connect:Direct for z/OS.	*name* | *generic* | (*list*) name—Specifies the Process name, up to 8 alphanumeric characters. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Locate the Process to select by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted.	*number from 1–99,999* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
queue	Specifies the Processes to be selected by the specified queue names. The default is **all**.	all | exec | hold | wait | timer all—Selects Processes from all queues. this is the default. exec—Selects Processes from the Execution queue. hold—Selects Processes from the Hold queue. timer—Selects Processes from the Timer queue. wait—Selects Processes from the Wait queue.
snode	Locate the Process by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
status	Specifies the Processes to be selected by Process status. If you do not specify a status value, information is generated for all status values.	EX | HC | HE | HI | HO | HR | HS | PE | WC | WR | WS | (*list*) EX (Execution)—Specifies to select Processes from the Execution queue. HC (Held for Call)—Specifies to select Processes submitted with hold=call. HE (Held due to Error)—Specifies to select Processes held due to a connection error. HI (Held Initially)—Specifies to select Processes submitted with hold=yes. HO (Held by Operator)—Specifies to select Processes held by a **change process** command issued with hold=yes. HR (Held Retain)—Specifies to select Processes submitted with retain=yes or retain=initial. HS (Held Due to Execution Suspension)—Specifies to select Processes suspended by a flush process command issued with hold=yes. PE (Pending Execution)—Specifies to select Processes submitted with a **maxdelay** parameter and assigned PE status by the Process Manager just before a Session Manager is created to execute the Process. After the Session Manager initializes, the Process is placed on the Execution queue and the status is changed to EX. WC (Waiting for Connection)—Specifies to select Processes that are ready for execution, but that all available connections to the remote node are in use. WR (Waiting for Restart)—Specifies to select Processes that are waiting for restart after session failure. WS (Waiting for Start Time)—Specifies to select Processes waiting for a start time. These Processes are on the Timer Queue. list—Specifies a list of status values. Enclose the list in parentheses, and separate each value with a comma.

Parameter	Description	Value
submitter	Locate the Processes to select by the node specification (the Sterling Connect:Direct node name) and user ID of the Process owner. The length of this parameter is unlimited.	(*node specification*, *userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification (the Sterling Connect:Direct node name) and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.
detail	Specifies the type of report (short or detailed) that Sterling Connect:Direct generates for the selected Processes.	yes | no yes—Generates a detailed report. no—Generates a short report. This is the default.

The following command displays a short report for the specified Process number:

```
select process pnumber=9 detail=no;
```

Output from the command is displayed in the following table:

```
================================================================================
                            SELECT  PROCESS
================================================================================
PROCESS NAME NUMBER  USER    SUBMITTER NODE   QUEUE        STATUS
--------------------------------------------------------------------------------
PR01     9        root    cd.unix.pj         EXEC     EX
================================================================================
```

The following command displays a detailed report for the specified Process number:

```
select process pnumber=9 detail=yes;
```

Output from the command is displayed in the following table:

```
================================================================================
             SELECT PROCESS
================================================================================
Process Name    => pr01           Class         => 9
Process Number  => 9              Priority      => 8
Submitter Node  => cd.unix.pj     PNODE         => cd.unix.pj
Submitter       => sub1           SNODE         => cd.unix.pj
Retain Process  => no             Header Type   => p

Submit Time     => 19:52:35       Schedule Time =>
Submit Date     => 05/22/1996     Schedule Date =>

Queue           => EXEC
Process Status  => EX
Message Text    =>
--------------------------------------------------------------------------------
```

Determining the Outcome of a Process

The **select statistics** command is used to examine Process statistics from the Sterling Connect:Direct statistics file. The type of information in the statistics report includes copy status and execution events.

The search criteria provide flexibility in selecting information you want to display. The parameters used with the select statistics command determine search criteria and the form in which the information is presented. You can specify records to select by condition code, Process name, Process number, identification type, category, secondary node, start time, stop time, and submitter node specification and user ID.

There are no required parameters for this command. If you do not indicate a search requirement with an optional parameter, Sterling Connect:Direct selects all statistics records; however, the volume of records can be excessive. Following are parameters for the select statistics command:

Parameter	Description	Value
ccode	Selects statistics records based on the completion code operator and return code values associated with Step Termination. You must specify the return code.	operator, *nn* operator—Specifies the completion code operator. Following are the valid completion code operators: eq or = or == (equal) This is the default. ge or >= or => (greater than or equal) gt or > (greater than) le or <= or =< (less than or equal) lt or < (less than) ne or != (not equal) The return code is the exit status of the UNIX command or the Sterling Connect:Direct Process or command. nn—Specifies the return code value associated with Step Termination.
destfile	Selects statistics based on a destination file name. This parameter can be abbreviated as dest.	dest=*/path/file name* For example: sel stat dest=/sci/payroll/june.payroll; This parameter can be used in combination with the srcfile parameter to select statistics based on a source file name and a destination file name, for example: sel stat srcf=/sci/accounting/june.payroll dest=/sci/payroll/june.payroll

Parameter	Description	Value
pname	Locate the statistics to select by Process name. The Process name is limited to 8 characters on Sterling Connect:Direct for Microsoft Windows and Sterling Connect:Direct for z/OS.	*name* | *generic* | (*list*) name—Specifies the Process name, up to 8 alphanumeric characters. generic—Specifies a nonspecific value for the Process name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more pname strings. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma.
pnumber	Locate the statistics to select by Process number. Sterling Connect:Direct assigns the Process number when the Process is submitted.	*number from 1–99,999* | (*list*) number—Specifies the Process number. list—Specifies a list of Process numbers. Enclose the list in parentheses, and separate each value with a comma.
reccat	Specifies whether the selection of statistics file records is based on events or related to a Process.	CAEV | CAPR | (CAEV, CAPR) CAEV—Specifies that the selection of statistics file records is related to an event, such as a Sterling Connect:Direct shutdown. CAPR—Specifies that the selection of statistics file records is related to one or more Sterling Connect:Direct Processes.
recids	Specifies the statistics file records to be selected by record ID. This parameter identifies particular types of statistics records, such as a copy termination records or Sterling Connect:Direct initialization event records.	*record id* | (*list*) record id—Selects statistics file records for the specified record ID. list—Specifies a list of Process names. Enclose the list in parentheses, and separate each value with a comma. Following are the valid record ID values: APSM—License Management failure generated. CHGP—The **change process** command issued. CRHT—Copyright statement. COAC—Listen connection enabled for either API or a remote node. CSPA—Sterling Connect:Direct Secure Plus failure generated. CSTP—Child Process stopped. CTRC—Copy termination record.

Parameter	Description	Value
		CTRM—Child Process terminated. CUKN—Child Process unknown status. CXIT—Child Process exited. DELP—The delete Process command issued. FLSP—The flush Process command issued. FMRV—Error occurred in function management. information receive operation. FMSD—Error occurred in function management. information send operation. GPRC—Error occurred while getting Process. IFED—The if statement ended. LSST—The record ID of a step on the local node. LIEX—License expired. LWEX—License expires within 14 days. NINF—Sterling Connect:Direct information generated at startup. NMOP—Network map file opened. NMPR—The network map is updated through Sterling Connect:Direct Browser User Interface, Sterling Control Center Console, or KQV Interface. NUIC—Initialization complete. NUIS—Initialization started. NUTC—Termination completed. NUTS—Termination started. PERR—Process error. PFLS—Process flushed. PRED—Process ended. PRIN—Process interrupted. PSAV—Process saved.

Parameter	Description	Value
		PSTR—Process started. QCEX—A Process moved from another queue to the EXEC queue. QCWA—A Process moved from another queue to the WAIT queue. QCTI—A Process moved from another queue to the TIMER queue. QCHO—A Process moved from another queue to the HOLD queue. RJED—The run job ended. RNCF—Remote node connection failed. RSST—The record ID of a step on the remote node. RTED—The run task ended. RTSY—Run task restarted. Resyncing with run task that was executing. SBED—The submit ended. SELP—The select Process command issued. SELS—The select statistics command issued. SEND—Session ended. SERR—System error. SFSZ—Size of the file submitted. SGON—User signed on using KQV Interface or Command Line Interface. SHUD—Shutdown occurred. SIGC—Signal caught. SSTR—Session start. STOP—The stop command issued. SUBP—The submit command issued. TRAC—The trace command issued. TZDI—The time zone of the local node represented as the difference in seconds between the time at the local node and the Coordinated Universal Time.

Parameter	Description	Value
		UNKN—Unknown command issued. USEC—Security check for user ID failed. USMG—Sterling Connect:Direct is shutting down. XCMM—Command manager (CMGR) messages. XCPR—Copy receive. XCPS—Copy send. XIPT—Communication errors. XLKL—Low-level TCQ record locking errors. XMSG—Message sent to user exit. XPAE—Parsing error occurred when a Process or command was submitted. XPAM—Parsing error occurred when a Process or command was submitted. XPMC—Process manager (PMGR) connection error messages.XPML—PMGR statistics log error messages. XPMP—PMGR error messages when checking permission on the Sterling Connect:Direct programs. XPMR—PMGR RPC and miscellaneous error messages. XPMT—PMGR termination error messages. XRPM—Run task or run job error messages. XRRF—Relative record file access error messages. File structure is use for TCQ. XSMG—Session manager (SMGR) error messages. XSQF—File access error messages. XSTA—User exit program started. XTQG—A single TCQ error message group. XTQZ—A single TCQ error message group.

Parameter	Description	Value
snode	Locate the statistics file record by the secondary node name. This parameter can be used to specify a specific remote node, a generic value for matching remote node names (using pattern matching), or a list of multiple remote node names. The secondary node name typically contains the 1–16 character remote Sterling Connect:Direct node name, but can be any string up to 256 alphanumeric characters long. You can also specify a remote node name as an IP address or hostname and a port number.	*remote node specification* | *generic* | (*list*) remote node specification—Identifies a specific remote node name. generic—Specifies a nonspecific value for the remote node name. This generic value, containing pattern-matching characters, evaluates to a list of zero or more remote node names. list—Specifies a list of remote node specifications. Enclose the list in parentheses, and separate each value with a comma.
srcfile	Selects statistics based on a source file name. This parameter can be abbreviated as srcf.	srcf=*/path/file name* For example: sel stat srcf=/sci/accounting/june.payroll; This parameter can be used in combination with the destfile parameter to select statistics based on a source file name and a destination file name, for example: sel stat srcf=/sci/accounting/june.payroll dest=/sci/payroll/june.payroll
startt	Selects records produced both at and since the specified time. The date or day and time are positional. If you do not specify date or day, a comma must precede time.	[*date* | *day*] [, *hh:mm:ss* [am|pm]] date—Specifies the day (dd), month (mm), and year (yy), which you can code as mm/dd/yyyy or mm-dd-yyyy. If you only specify date, the time defaults to 00:00:00. The current date is the default. day—Specifies the day of the week. Values are today, monday, tuesday, wednesday, thursday, friday, saturday, and sunday. hh:mm:ss [am | pm]—Specifies the time of day in hours (hh), minutes (mm), and seconds (ss). You can specify the hour in either 12- or 24-hour format. If you use 12-hour format, then you must specify am or pm. The default is the 24-hour format. The default value is **00:00:00**, which indicates midnight. If you specify only the day value, the time defaults to 00:00:00.

Parameter	Description	Value
stopt	Specifies that Sterling Connect:Direct searches for statistics records up to and including the designated date, day, and time positional parameters. If you do not specify date or day, a comma must precede time.	[*date* | *day*] [, hh:mm:ss [am|pm]] date—Specifies the day (dd), month (mm), and year (yy), which you can code as mm/dd/yyyy or mm-dd-yyyy. If you only specify date, the time defaults to 00:00:00. The current date is the default. day—Specifies the day of the week. Values are today, monday, tuesday, wednesday, thursday, friday, saturday, and sunday. hh:mm:ss [am | pm]—Specifies the time of day in hours (hh), minutes (mm), and seconds (ss). You can specify the hour in either 12- or 24-hour format. If you use 12-hour format, then you must specify am or pm. The default is the 24-hour format. The default value is **00:00:00**, which indicates midnight. If you specify only the day value, the time defaults to 00:00:00.
submitter	Locate the statistics records to select by the node specification (the Sterling Connect:Direct node name) and user ID of the Process owner. The character length of the parameter is unlimited.	(*node specification, userid*) | *generic* | (*list*) node specification, userid—Specifies the node specification (the Sterling Connect:Direct node name) and user ID. generic—Specifies a nonspecific value for node specification and user ID. This generic value, containing pattern-matching characters, evaluates to a list of zero or more node specifications and user IDs. list—Specifies a list of node specification and user ID pairs. Enclose the list in parentheses, and separate each value with a comma.
detail	Specifies the type of report (short or detailed) that Sterling Connect:Direct generates for the selected Processes.	yes | no yes—Generates a detailed report. no—Generates a short report. This is the default.

Generating a Detailed Output Report for a Process

You can use the **select statistics** command to generate a detailed report for a Process. The following command generates a detailed report for Process number 9:

```
select statistics pnumber=9 detail=yes startt=(08/10/2008);
```

The report consists of all records from August 10, 2008.

A sample statistics output for two steps only is listed in the following section. Use the table of recids in “Determining the Outcome of a Process” on page 26 to interpret the Record ID. The Record ID can change for each Process step displayed. The completion code indicates whether the Process executed successfully or produced an error condition.

To display the long text of the message, issue the **ndmmsg** command.

Generating a Summary Report for a Process

You can use the **select statistics** command to generate a summary report for a Process. The following command generates summary statistics for Process number 9:

```
sel stat pnumber=9 detail=no startt=(08/10/2008);
```

The report consists of all records from August 10, 2008.

Sample output that describes all Process steps in summary form is displayed in the following table:

```
================================================================================
                          SELECT STATISTICS
================================================================================
P RECID LOG TIME             PNAME PNUMBER STEPNAME CCOD FDBK MSGID
--------------------------------------------------------------------------------
P PSTR 08/10/2008 09:10:39   PR01  9                0         XSMG200I
P IFED 08/10/2008 09:10:44   PR01  9                0         XSMG405I
P CTRC 08/10/2008 09:10:44   PR01  9                0         XSMG405I
P IFED 08/10/2008 09:10:45   PR01  9                4         XSMG400I
P RTED 08/10/2008 09:10:45   PR01  9                0         XSMG400I
P IFED 08/10/2008 09:10:45   PR01  9                4         XSMG400I
P CTRC 08/10/2008 09:10:45   PR01  9                0         XSMG405I
P CTRC 08/10/2008 09:10:45   PR01  9                8         XSMG405I
P CTRC 08/10/2008 09:10:45   PR01  9                8         XSMG405I
================================================================================
```

To avoid lengthy search times when issuing the **select statistics** command, archive or delete statistics files regularly. Also, use the **startt** and **stopt** parameters to bracket the desired stats as closely as possible. Execution of a Process generates multiple statistics records. Sterling Connect:Direct closes the current statistics file and creates a new statistics file every midnight. It can also close the current file before midnight if the file size exceeds the value set for the file.size initialization parameter. The default file size is 1 megabyte.

Statistics files are in the *d_dir/work/cd_node* directory. Names of the statistics file are in the format **Syyyymmdd.ext**, where **yyyy** indicates year, **mm** indicates month, and **dd** indicates day. The extension (ext) begins as 001. The extension is incremented by one each time a new statistics file is created in a single day.

Running System Diagnostics

The diagnostic command, trace, enables you to run system diagnostics and troubleshoot operational problems. Use the **trace** command with the appropriate parameter listed in the following table to enable and disable runtime traces within the Process Manager, Command Manager, and Session Manager components of the software. For Session Manager traces, you can run a trace for a specific node.

The Command Manager trace is turned on immediately for the client that issued the trace command. After the trace command is issued, all clients that make connections are also traced. Session Manager traces go into effect immediately.

The **trace** command has the following parameters:

Parameter	Description	Value
cmgr	To trace the Command Manager.	level=0 |1 | 2 | 4 file=name level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—Is the basic level that provides function entry and function exit.⌂2—Includes level 1 plus function arguments.⌂4—Enables a full trace. Basic diagnostic information, such as values of internal data structures at key points in the execution flow, are displayed. file—Specifies the name of a file where the trace output is directed. If you do not specify a file name, the file is created in the Sterling Connect:Direct working directory with the file name **CMGR.TRC**. The length of the name value is unlimited.
comm	To trace the data sent to and received from a remote Sterling Connect:Direct system within the Session Manager. You can set this trace independently from or in conjunction with the smgr trace. If you run both the comm and smgr traces, trace output for both traces is directed to the file name of the trace last specified.	level=0 |1 | 2 | 4 file=name level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—Is the basic level that provides function entry and function exit.⌂2—Includes level 1 plus function arguments.⌂4—Enables a full trace that provides basic diagnostic information, such as values of internal data structures at key points in the execution flow. file—Specifies the name of a file where the trace output is directed. If you do not specify a file name, the file is created in the Sterling Connect:Direct working directory with the file name COMM.TRC. The length of the name value is unlimited. The default file name is **COMM.TRC**.

Parameter	Description	Value
pmgr	To trace the Process Manager.	level=0 |1 | 2 | 4 file=name level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—Is the basic level that provides function entry and function exit.⌂2—Includes level 1 plus function arguments.⌂4—Enables a full trace that provides basic diagnostic information, such as values of internal data structures at key points in the execution flow. file—Specifies the name of a file where the trace output is directed. If you do not specify a file name, the file is created in the Sterling Connect:Direct working directory with the file name **PMGR.TRC**. The length of the name value is unlimited.
smgr	To run the trace for Session Managers created after issuing the trace command. Currently executing Session Managers are unaffected. If you run both the comm and smgr traces, trace output for both traces is directed to the file name of the trace last specified.	level=0 |1 | 2 | 4 snode | pnode | tnode file=name level—Specifies the level of detail displayed in the trace output. The default is **4**. 0—Terminates the trace.⌂1—Is the basic level that provides function entry and function exit.⌂2—Includes level 1 plus function arguments.⌂4—Enables a full trace that provides basic diagnostic information, such as values of internal data structures at key points in the execution flow. snode—Specifies to trace only the SNODE SMGR. pnode—Specifies to trace only the PNODE SMGR. tnode—Identifies the node on which to perform the trace. If you want to gather trace information for more than one node, identify more than one node in this parameter. file—Specifies the name of a file where the trace output is directed. If you do not specify a file name, the file is created in the Sterling Connect:Direct working directory with the file name SMGR.TRC. The length of the name value is unlimited. The default file name is **SMGR.TRC**.

The following sample trace command performs a level 2 trace on the Session Manager for the node called ath3500ry and writes the output to the file Smgp.trc:

```
trace smgr pnode tnode=ath3500ry level=2 file=Smgp.trc;
```

A partial sample trace output is illustrated in the following section. A trace identifies the Process ID and the function, the month and day, and the time in microseconds. The first column contains the Process ID. Column two indicates the month and day in the form of MM/DD. Column three indicates the time in the form of HH:MM:SSSS. The last column indicates the function. An arrow pointing to the right indicates the function was entered. An arrow pointing to the left indicates the function was exited. Some of the functions are indented, which indicates nesting. An indented arrow indicates that the function was called by the preceding function.

```
indicates that the function was called by the preceding function.
================================================================================
498    05/18 15:13:0104   cm_sendcmd_1 entered.
498    05/18 15:13:0206    -> ndm_error_destroy
                         <- ndm_error_destroy: ok
498    05/18 15:13:0506    -> ndm_error_create
                         <- ndm_error_create: ok
498    05/18 15:13:0708    ndm_cmds_free entered.
                         ndm_cmds_free exited.
498    05/18 15:13:0801    ->ndm_parser_jdi
498    05/18 15:13:0806    -> ndm_error_create
                          <- ndm_error_create: ok
498    05/18 15:13:0916     ->Parser: SELPRO
498    05/18 15:13:0926      ->bldexp
                           <-bldexp: Null argument value,
                                   don&apos;t add.
498    05/18 15:13:1116      ->bldexp
498    05/18 15:13:1136       -> ndm_crit_comp
498    05/18 15:13:1155        ->compile
                             <-compile
                            <- ndm_crit_comp: Handle
                           <-bldexp: ok
                       .
                       .
                       .
================================================================================
```

Chapter 2. Process Queuing

Overview of the Transmission Control Queue

The TCQ controls Process execution as Sterling Connect:Direct operates. After you submit a Process, it is stored in the TCQ. The TCQ consists of four queues: Execution, Wait, Timer, and Hold.

After you submit a Process, you can monitor the status, modify specific characteristics, and stop execution by using the appropriate commands. The commands listed in the following table allow you to perform these tasks:

Command	Definition
change process	Change the status and modify specific characteristics of a nonexecuting Process in the TCQ.
delete process	Remove a nonexecuting Process from the Wait, Timer, and Hold queues.
flush process	Remove an executing Process from the Execution queue.
select process	Monitor Processes in the TCQ, including those Processes that are executing.
view process	View Processes in the TCQ.

Scheduling Sterling Connect:Direct Activity

Sterling Connect:Direct places a Process in a queue based on the parameters that affect scheduling. You can specify scheduling parameters in the **Process** statement or the **submit** command.

Scheduling parameters are listed in the following section:

- retain=yes | no | initial
- hold=yes | no | call
- startt=[([date | day] [, hh:mm:ss | [am | pm]])

The following table shows how scheduling parameters affect the logical queues.

Scheduling Parameter	Queue	Comments
None of the scheduling parameters specified	Wait	The Process remains in the Wait queue until Sterling Connect:Direct establishes a session with the remote node. After a session is established, the Process moves to the Execution queue.
retain=yes	Hold	A copy of the Process executes once, unless you specify a startt parameter value. Specify a day or time or both for the Process to start.
retain=no	Wait (if no other parameters are specified)	The Process remains in the Wait queue until Sterling Connect:Direct establishes a session with the remote node. The default is **no**.

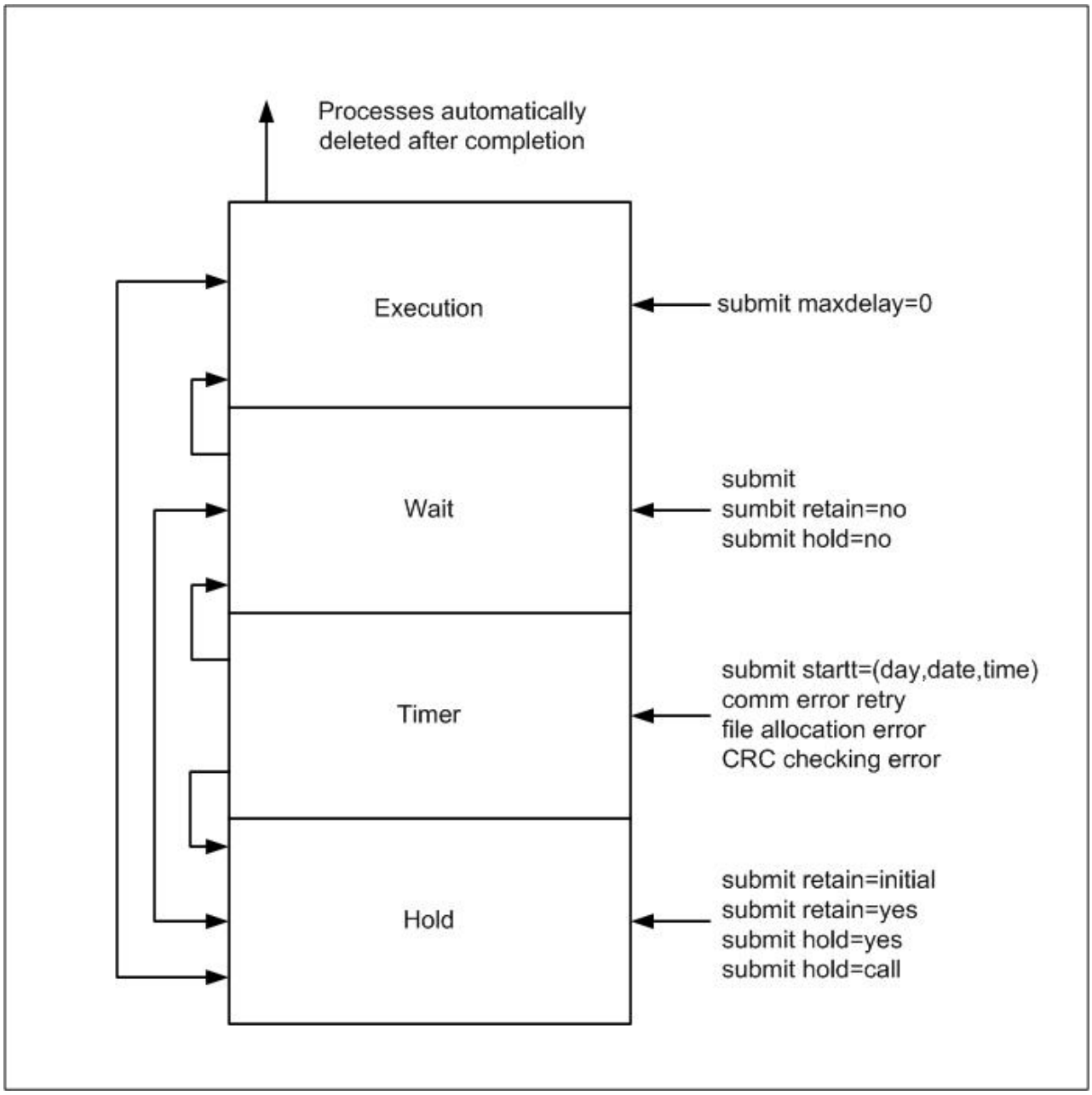
Scheduling Parameter	Queue	Comments
retain=initial	Hold	A copy of the Process remains in the Hold queue and executes every time the Process Manager is initiated.
retain=yes and hold=no or hold=call	Hold	A copy of the Process remains in the Hold queue to be executed when released.
hold=yes	Hold	You can execute the Process by specifying the **change process** command with the release parameter.
hold=no	Wait (if no other parameters are specified)	The default for hold is **no**.
hold=call	Hold	The Process remains in the queue until the remote node starts a session with the local node or another Process starts a session with that remote node.
startt	Timer	When the scheduled day or time occur, the Process is moved to the Wait queue.

Each Process in the TCQ has an associated status value. Each status value has a unique meaning that is affected by the logical queue in which the Process is placed. Status values for each queue are shown in the tables in the following sections. You can use the **select process** command to examine that status of Processes in the TCQ. For example, the following command displays all Processes in the TCQ with execution status:

```
select process status=EX;
```

Progression of a Process Through the TCQ

This section describes each logical queue of the TCQ and the progression of a Process through these queues. The following figure illustrates the four logical queues and their associated parameter values:



The Execution Queue

Processes are placed in the Execution queue after Sterling Connect:Direct connects to the remote node. Processes normally come from the Wait queue, but also can be placed in the Execution queue by a submit command with maxdelay=0 specified.

Processes in the Execution queue can be in execution (EX) status or pending execution (PE) status. Processes with EX status are exchanging data between two Sterling Connect:Direct nodes. Processes with PE status are waiting for Process start messages to be exchanged between the local node and the remote node. Processes usually have PE status assigned for a very short period of time.

After a Process successfully completes, it is automatically deleted from the Execution queue. A flush process command with hold=yes moves a Process from the Execution queue and places it in the Hold queue. When a session is interrupted, the Process moves from the Execution queue to the Timer queue if retry values are specified. If connection is not made before the retry values are exhausted or if retry values are not specified, the action taken depends on the **conn.retry.exhaust.action** parameter. By default, the Process moves to the Hold queue.

The following table shows the status values for the Execution queue:

Element	Comment
PE	Pending Execution is the initial queue status when a Process is submitted with maxdelay=0.
EX	Execution status indicates that the Process is executing.

The Wait Queue

Processes in the Wait queue are waiting for a new or existing connection to become available between the local node and the remote node.

Processes can come from the Hold queue or the Timer queue. Processes also can be placed in the Wait queue by a submit command with no parameters specified, submit with retain=no, or submit with hold=no.

After the connection is made, Processes automatically move to the Execution queue.

The following table shows the status values for the Wait queue:

Status	Comment
WC	This status indicates the Process is ready to execute as soon as possible, but no session is available. Other Processes may be executing with the SNODE, and no other sessions are available. This Process runs as soon as a new session is created or an existing session becomes available.
WR	This status indicates that the Process is in retry status. The number of retries and intervals between retries is specified in the network map.
WA	This status indicates the initial queue status when a Process is submitted without a hold or retain value. This Process is ready to execute as soon as possible.
WS	This status indicates that the Process is waiting for the PNODE to continue the session.

The Timer Queue

Processes are placed in the Timer queue by a submit command with the startt parameter specified. Processes in the Wait for Start Time (WS) status are waiting for the start time to arrive before moving to the Wait queue. Processes also are placed in the Timer queue in Retry (WC) status if one of the following error conditions occur:

- If a file allocation error occurs when a Process is executing on either the local or the remote node, and the file allocation error is identified as a condition to retry, the Process is placed in the Timer queue. The Process is then retried using the short-term and long-term retry parameter definitions. This capability enables a Process that was unable to execute because a file that it called was unavailable to be retried at a later time.
- If a connection error occurs while a Process is executing, the intelligent session retry facility places all Processes scheduled for the node, including the executing Process, in the Timer queue. This capability eliminates the overhead required to retry each of the Processes on the node even though the connection is lost.
- If CRC checking is activated, a Process that generates a CRC error is placed in the Timer queue.

Sterling Connect:Direct automatically tries to execute the Process again based on the number of times to retry and the delay between retries as specified in the network map parameters.

Processes move from the Timer queue to the Wait queue. A **change process** command with hold=yes specified moves the specified Process from the Timer queue to the Hold queue. The following table shows the status values for the Timer queue:

Status	Comment
WR	This status indicates that the Process is in retry status. The number of retries and intervals between retries is specified in the network map.
WS	This status indicates that the Process is waiting for the PNODE to continue the session.
HR	Held Retain indicates that the Process was submitted with retain=yes or retain=initial specified and has already executed. The Process can be released later by a **change process** command with release specified.
WC	This status indicates the Process is ready to execute as soon as possible, but no session is available. Other Processes may be executing with the SNODE, and no other sessions are available. This Process runs as soon as a new session is created or an existing session becomes available.

The Hold Queue

Processes in the Hold queue are waiting for operator intervention before they progress to the Wait queue. This queue enables operators of the local node and remote node to coordinate and control Process execution.

Processes are placed in the Hold queue by a submit command with retain=initial, retain=yes, or hold=yes parameters specified. Processes submitted with hold=call also are placed in the Hold queue. Processes are moved from the Timer queue to the Hold queue by a **change process** command with hold=yes specified. Additionally, Processes are moved from the Execution queue to the Hold queue by a f**lush process** command with hold=yes specified.

Processes are moved from the Hold queue to the Execution queue by a **change process** command with the release parameter specified.

The following table shows the status values for the Hold queue:

Status	Comment
HC	Held for Call indicates that the Process was submitted with hold=call specified. A session started from either node causes the Process to be moved to the Wait queue in WC status. The Process is placed in the Execution queue when the Process is selected for execution.
HI	Held Initially indicates that the Process was submitted with hold=yes specified. The Process can be released later by a **change process** command with release or hold=no specified.
HE	Held due to error specifies that a session error or other abnormal condition occurred.
HO	Held by Operator indicates that a change process hold=yes was specified.
HR	Held Retain indicates that the Process was submitted with retain=yes or retain=initial specified and has already executed. The Process can be released later by a **change process** command with release specified.
HS	Held for Suspension indicates that the operator issued a **flush process** command with hold=yes specified. The Process can be released later by a **change process** command with release specified.

Chapter 3. Sterling Connect:Direct Utilities

Introduction to Translation Tables

Sterling Connect:Direct translates data from one character set code to a different character set code, such as from ASCII to EBCDIC, based on a character translation table in the *d_dir*/ndm/xlate directory. Sterling Connect:Direct provides a default character translation table for use during file transfer operations or you can modify this table using the utility program called ndmxlt.

Creating a Translation Table

1. To create a translation table, either copy the file called */cd_dir*/cdunix/ndm/src/def_send.sxlt or */cd_dir*/cdunix/ndm/src/def_recv.sxlt, where *cd_dir* is the directory where Sterling Connect:Direct is installed, and rename it or modify this file.
2. Use a text editor to add the new values to the table in the file you created.
3. Compile the updated file with the ndmxlt utility.
4. Replace the default translation table in the *d_dir*/ndm/xlate with the updated table. Each table is 256 bytes long.

```
Following is a sample translation table:
# This file contains an example of defining an ASCII-to-EBCDIC translation table and
# then changing it to translate lowercase to uppercase.
#
# Define the ASCII-to-EBCDIC table.
offset=0
00 01 02 03  04 05 06 07 08 05 15 0B 0C 0D 0E 0F
10 11 12 13  3C 15 16 17 18 19 1A 1B 1C 1D 1E 1F
40 5A 7F 7B  5B 6C 50 7D 4D 5D 5C 4E 6B 60 4B 61
F0 F1 F2 F3  F4 F5 F6 F7 F8 F9 7A 5E 4C 7E 6E 6F
7C C1 C2 C3  C4 C5 C6 C7 C8 C9 D1 D2 D3 D4 D5 D6
D7 D8 D9 E2  E3 E4 E5 E6 E7 E8 E9 AD E0 BD 5F 6D
79 81 82 83  84 85 86 87 88 89 91 92 93 94 95 96
97 98 99 A2  A3 A4 A5 A6 A7 A8 A9 C0 4F D0 A1 7F
80 81 82 83  84 85 86 87 88 89 8A 8B 8C 8D 8E 8F
90 91 92 93  94 95 96 97 98 99 9A 9B 9C 9D 9E 9F
A0 A1 A2 A3  A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF
B0 B1 B2 B3  B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF
C0 C1 C2 C3  C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF
D0 D1 D2 D3  D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF
E0 E1 E2 E3  E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF
F0 F1 F2 F3  F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF
#
# Change the lowercase characters to uppercase.
offset=61
C1 C2 C3 C4  C5 C6 C7 C8 C9 D1 D2 D3 D4 D5 D6 D7
D8 D9 E2 E3  E4 E5 E6 E7 E8 E9
```

Each byte stores the character value for the target character set. The source character set is used as an index into the table. For example, an ASCII blank (Hex 20) would locate the byte at offset Hex 20 in the translation table. If the byte at location Hex 20 contains Hex code 40, that would translate to an EBCDIC code indicating a blank character.

Compiling a Translation Table Using the ndmxlt Utility

Before you begin

You can create or modify a translation table tailored to your requirements with the ndmxlt utility program.

To invoke the **ndmxlt** utility, type the following command at the UNIX prompt:

```
$ ndmxlt -ssourcefile -ooutputfile [ -rradix] [ -ffiller] -mxlatefile
```

The parameters for the ndmxlt command are listed in the following table:

Parameter	Description	Values
-ssourcefile	The path and file name of the translation table source file. If no value is specified, input is read from STDIN.	Path and name of translation table
-ooutputfile	The path and file name of the translation table output file.	Path and name of translation output file
-rradix	The radix or base of the source file input data. All numeric values whether from command line options or input data are interpreted based on the radix setting.	x | d | o x—Hexadecimal. This is the default. d—Decimal o—Octal The default is x.
-ffiller	A filler byte value. The entire table is initialized to this value before the input data is scanned and applied to the table.	Any keyboard character, number, or special character, plus control characters entered using a preceding slash. For example, "\0" is null.
-m	The path and file name of a model translation table. If specified, the model table is read in and then the input data is scanned and applied to the table. This capability permits creating a number of different tables that are variations from a single base table without having to specify all 256 bytes of input data for each table.	Path and file name of the model translation table

Example—Creating a Translation Table

About this task

Perform the following steps to create a sample translation table that changes lowercase characters to uppercase characters:

Procedure

1. Make a copy of the sample translation table located at cd_dir/ndm/src/def_send.sxlt.
2. Open the new translation table with a text editor.

3. Add the following lines to the bottom of the table. It should look like the table in “Creating a Translation Table” on page 43when you have added this information.

```
#
# Change the lowercase characters to uppercase.
offset=61
C1 C2 C3 C4 C5 C6 C7 C8 C9 D1 D2 D3 D4 D5 D6 D7
D8 D9 E2 E3 E4 E5 E6 E7 E8 E9
```

4. Copy the modified file to *cd_dir*/ndm/src and name it UpperCaseEBC.sxlt.
5. Compile the new translation table using the following syntax:

```
ndmxlt -s../src/UpperCaseEBC.sxlt -oUpperCaseEBC.xlt
```

6. To use this translation table, add the following sysopts parameter to the copy statement:

```
copy from file=filename
     to   file=filename
          sysopts=":xlate.tbl=pathname/UpperCaseEBC.xlt:"
```

Example—Modifying a Model Translation Table

About this task

Perform the following steps to modify a model translation table. This method, when implemented, reads the model table and writes it to a new file. It then reads the input data and makes changes to the table created.

Procedure

1. Create a file called FourLinesUpperCase.sxlt and add the following lines to the file:

```
#
# Change the lowercase characters to uppercase.
offset=61
C1 C2 C3 C4 C5 C6 C7 C8 C9 D1 D2 D3 D4 D5 D6 D7
D8 D9 E2 E3 E4 E5 E6 E7 E8 E9
```

2. Copy the modified file to *cd_dir*/ndm/src.
3. Type the following command to compile this file and create a translation table called fourLineUpperCase.xlt:

```
ndmxlt -s../src/FourLineUpperCase.sxlt -oFourLineUpperCase.xlt -mdef_send.xlt
```

4. To use this translation table, add the following sysopts parameter to the copy statement:

```
copy from file=filename
     to   file=filename
          sysopts=":xlate.tbl=pathname/FourLineUpperCase.xlt:"
```

Using Translation During File Transfer Operations

Translation is specified in the copy statement of a Sterling Connect:Direct Process. You can use the default translation table or create a new table.

Translation is specified in the copy statement of a Sterling Connect:Direct Process. You can use the default translation table or create a new table.

To use the default translation table, type the following copy statement:

```
copy from file=abc
     to   file=xyz
          sysopts=":xlate.tbl=yes:"
```

To specify a customized table for data translation, include the following sysopts subparameter in the copy statement, where *pathname/filename* identifies the translation table:

```
copy from file=filename
     to   file=filename
          sysopts=":xlate.tbl=pathname/filename:"
```

Refer to the UNIX section of the IBM® Sterling Connect:Direct Processes Web site at http://www.sterlingcommerce.com/documentation/processes/processhome.html for additional details concerning translation table specification with a **copy** statement.

Translation Table Error Messages

The following table displays the error messages that are generated by **ndmxlt**:

Diagnostic Number	Description
XXLT001I	Invalid directive
XXLT002I	Input file open error
XXLT003I	Model file open error
XXLT004I	Invalid filler value
XXLT005I	Invalid offset value
XXLT006I	Invalid radix value
XXLT007I	Invalid table value
XXLT008I	Table data out of bounds

Accessing Sterling Connect:Direct Messages

The Sterling Connect:Direct message file contains records with text for all messages, including errors and messages from Sterling Connect:Direct servers other than the host server. You can add and delete message records with a text editor. The message file resides in *d_dir*/ndm/cfg/cd_node/msgfile.cfg. You can display message text with the **ndmmsg** command.

Message File Content

The message file is structured much the same way as other Sterling Connect:Direct configuration files. Each record is a logical line in a text file that consists of one or more physical lines. Each record has a unique name, a message ID, and fields that make up the message text.

The message record definitions provide for symbolic substitution, which permits including actual file names or other variable information within the text to more specifically identify a problem. Symbolic variables begin with the ampersand character (&).

The format of Sterling Connect:Direct message IDs is listed in the following table:

```
XxxxnnnI

Where:

X Indicates Sterling Connect:Direct
```

```
xxx is a 3-character Sterling Connect:Direct component identifier
nnn is a 3-digit decimal number
I is the standard, though not required, suffix
```

Message File Record Format

The following example shows the format of the message file record. Each record can be up to 4K bytes long. Optional parameters and values are in brackets.

```
message id [long.text detailed message explanation] [mod.name issuing module name] short.text
message summary
```

Following are the parameters for the message file record:

Parameter	Description	Values
long.text	A string that explains the message in detail.	A text string
mod.name	The name of the source module issuing the message ID.	Source module name
short.text	A summary of the message. This field is required.	Summary message, up to 72 characters

The following example illustrates a sample message record for XCPS008I:

```
XCPS008I:\ :mod.name=NUSMCP00.C:\
:short.text=File is not VB datatype.:\
:long.text=File is not variable block. Change sysopts datatype to\
either binary or text to transfer this file.\
\nSYSTEM ACTION-> the copy step failed and CD processing\
continued with the next process step.\
\nRESPONSE-> change the sysopts datatype to either\
binary or text.:\
```

Displaying Message Text

Use the ndmmsg command to display text in the message file. You can display both short and long text.

The following command illustrates the format for ndmmsg:

```
ndmmsg -f msgfname [-l | -s] msgid1 [msgid2 [msgid3 [...]]]
```

Following are the parameters for the ndmmsg command. If you do not specify an **l** or **s** parameter, both short and long text are displayed.

Parameter	Description
-f	Specifies the name of the message file.
-l	Displays the long text of a message.
-z	Displays the short text of a message.

Following is a sample **ndmmsg** command:

```
ndmmsg -f /usr/ndmunix/msgfile.cfg XCMG000I
```

Output from the command is displayed in the following example:

```
rc=&rc
fdbk=&fdbk
mod.name=NUCMRG00.C
func.name=ndmapi_sendcmd
```

```
short.text=CMGR RPC call returns NULL
long.text=The ndmapi_sendcmd RPC call made by the API to the CMGR returns a
NULL pointer.There is probably an RPC error.ndm.action=None
user.action=First, check if the ndmcmgr is still running; it could have
been killed accidently.If so, then abort the current CLI and restart the
CLI.  If the same problem occurs again, try to increase the value of wait
time (if set) in the API configuration file (ndmapi.cfg).
```

Precompressing/Decompressing Files Using the Standalone Batch Compression Utility

The Standalone Batch Compression Utility (cdsacomp) enables you to precompress files and then transfer the precompressed files to remote Sterling Connect:Direct nodes using Sterling Connect:Direct Processes. You have the following options for decompressing the files. A file can either be:

- Decompressed as it is received by the remote node (available on all Sterling Connect:Direct platforms)
- Stored on the remote node and later decompressed offline using cdsacomp (available only on Sterling Connect:Direct and Sterling Connect:Direct for z/OS).

Because cdsacomp can be used offline, it allows you to allocate some of the overhead associated with compression to non-peak times. For example, if you need to send the same file to several remote nodes, use this utility so that the file is precompressed only one time. You can also use cdsacomp to determine how much compression can be achieved for a file without having to transmit the file.

The cdsacomp utility is located in the Sterling Connect:Direct /bin directory.

Special Considerations for Using the Standalone Batch Compression Utility

Consider the following when you are using cdsacomp to precompress files:

- If you precompress a file with the cdsacomp utility, then you cannot specify any compression options in your Sterling Connect:Direct Process when you copy that file.
- You cannot specify data transformations (xlate, codepage, strip blanks, and so on) when sending a precompressed file with :precompress=yes: sysopts (for on-the-fly decompression). The following transformation options are available:
  - -x
  - -p
  - -s
  - -a
- If you precompress a file with the cdsacomp utility on a Sterling Connect:Direct node, then you cannot specify a checkpoint interval in your Sterling Connect:Direct Process if you decompress the file as it is received by the remote node.
- When you are copying a precompressed file to z/OS without :precomp=yes: (for deferred decompression):
  - The Copy operation must specify DCB information for the destination file. The physical block size of the destination file on Sterling Connect:Direct for z/OS must match the logical block size of the precompressed source file on Sterling Connect:Direct for UNIX.
  - The logical block size of the source file defaults to 27920 unless overridden by the -b parameter.

Using the Standalone Batch Compression Utility

Before you begin

To invoke the standalone batch compression utility (cdsacomp), type the following command at a UNIX prompt:

```
cdsacomp
```

Following are the parameters for the cdsacomp utility:

Parameter	Description	Values
-m	Specify which mode to use: precompress or decompress. This argument is required.	compress | decompress The default is **compress**.
-i	Specify the input file to precompress or decompress. This argument is required.	*full or relative path of input file*
-o	Specify the output file to save. If the output file already exists, it is overwritten. This argument is required.	*full or relative path of output file*
-z	Use this option with "-m compress" to override default compression values. This argument is optional. When decompressing, the values used during compression are used.	*level, window, memory* level—**Compression level**. The range is 1–9. The default is **1**. 1—Provides the least compression, but is the fastest. 9—Provides the most compression, but is the slowest. window—The size of the compression window and history buffer. Increasing the window increases the compression, but uses more virtual memory. The range is 9–15. The default is **13**. memory—**The amount of virtual memory to allocate.** The range is 1–9. The default is **4.** 1—Uses the least virtual memory. 9—Uses the most virtual memory.
-x	Use this option to translate the file. If this parameter is not specified, the file is not translated. This parameter is mutually exclusive with -codepage.	*full path to translate table file* | *relative path to translate table file*

Parameter	Description	Values
-p	Use this option to specify codepages for file conversion. Default is no codepage translation. This parameter is mutually exclusive with -xlate.	*source codepage, destination codepage*
-d	Specify the datatype of the file. When you use "-m compress", the datatype values result in the following: • text Strips newline characters from each record Supports -s and -a parameters Uses data attributes of blocksize=23040, recfm=vb, lrecl=23036, dsorg=ps • binary Uses data attributes of blocksize=23040, recfm=u, lrecl=0, dsorg=ps Does not support -s and -a parameters • VB Does not support -x, -p, -s, and -a parameters Uses data attributes of blocksize=23040, recfm=vb, lrecl=23036, dsorg=ps When you use "-m decompress", the datatype values result in the following: • text Inserts newline characters into each record Supports the -s parameter • binary Does not support the -s parameter • VB Does not support -x, -p, and -s parameters	text | binary | VB The default is **text.**
-b	Specify the block size of the output file. This parameter is valid only when you specify ⌂"-m compress" for the compression option.	*nnnnn* The range is 4096–32760. The default is **27920**.
-s	Use this option to strip trailing blanks. This parameter is valid only when you specify ⌂"-d text" for the datatype of the file.	y | n y—yes n—no The default is **y**.

Parameter	Description	Values
-a	Use this option to replace zero-length records with a single, blank character. This parameter is valid only when you specify the following: “-d text” and “-m compress”.	y | n y—yes n—no The default is **y**. Specify **n** if the data is copied to an i5OS or mainframe node.
-h	Use this option to display online help for the utility.	No values are available for this parameter.

Example—Precompress a Text File

In this example, the source file is a text file named source.file which is precompressed into a destination file named compressed.file. The file is translated using the default translation table, /home/cd/ndm/xlate/def_send.xlt. Trailing blanks are stripped. Default settings for ZLIB tuning, checkpoint interval and block size are used.

```
cdsacomp -m compress
         -d text
         -i source.file
         -o compressed.file
         -x /home/cd/ndm/xlate/def_send.xlt
         -s y
```

Example—Precompress a Text File With Codepage Conversion

In this example, the source file is a text file named zzz.sac which is precompressed into a file named zzz.txt. The file is converted from EBCDIC-US to ASCII using the codepage option. Default settings are used for parameters that are not specified.

```
cdsacomp -m compress
         -d text
         -i zzz.txt
         -o zzz.sac
         -p EBCDIC-US,ASCII
```

Example—Precompress a Binary File

In this example, the source file is a binary file named source.file which is precompressed into a destination file named compressed.file. Default settings are used for parameters that are not specified.

```
cdsacomp -m compress
         -d binary
         -infile source.file
         -outfile compressed.file
```

Example—Decompress a Text File

In this example, the source file is a precompressed text file named compressed.file which is decompressed into a destination file named dest.file. The file is translated using the default translation table, /home/cd/ndm/xlate/def_recv.xlt. Default settings are used for parameters that are not specified.

```
cdsacomp -m decompress
         -d text
         -i compressed.file
         -o dest.file
         -x /home/cd/ndm/xlate/def_recv.xlt
```

Examples—csdacomp Command Help

Requesting a summary of cdsacomp command parameters and help options:

```
cdsacomp -h
```

Example—Decompress a File on the Remote Node During the Copy Step

The “precomp=yes” parameter is used when the file was compressed by the cdsacomp utility prior to this Process. The file is transferred by this Process as a pre-compressed file. It is then decompressed by special processing as it is received on the remote node.

```
sample process snode=cdunix1
step01 copy
from
(
file=/home/cd/upload/compressed.file
sysopts=":precomp=yes:"
pnode
)
to
(
file=/home/cd/download/decompressed.file
snode
disp=rpl
)
pend;
```

Example—Send Precompressed File to z/OS and Storing It as Precompressed

The precompressed file is copied to the z/OS node with PNODE sysopts of “datatype=binary”. The destination file is not decompressed. The DCB settings of the original precompressed file are preserved on the z/OS node. The specified checkpoint interval will be used during the file transfer. The file can be decompressed with the z/OS cdsacomp utility.

```
sample process snode=cdunix1
step01 copy
from
(
file=/home/cd/upload/compressed.file
sysopts=":datatype=binary:"
pnode
)
chkpt=2M
to
(
file=upload.compressed.file
dcb=(blksize=27920, lrecl=0, dsorg=ps, recfm=u)
snode
disp=(new,catlg)
)


  pend;
```

Validate Configuration Files

When you manually edit any of the five text-based Sterling Connect:Direct configuration files, the Configuration Checking Utility (cfgcheck) enables you to validate these files offline. The following files can be validated using this utility: userfile.cfg, initparm.cfg, netmap.cfg, ndmapi.cfg, and sysacl.cfg.

Note: The Strong Access Control File (sysacl.cfg) will be validated only when the user running the Configuration Checking Utility is a root user.

By default, cfgcheck is run with no arguments and attempts to find all five of the configuration files in the current working directory. If all of the Sterling Connect:Direct components are not installed, then some of the files will not be found. For example, if the Command Line Interface (CLI) is installed but the Sterling Connect:Direct server is not installed, only the ndmapi.cfg file will exist in the installation directory. Therefore, only the ndmapi.cfg file will be validated. When cfgcheck is run with no arguments, the utility will report that the other configuration files were not found.

Note: Before you can execute cfgcheck, you must set the NDMAPICFG environment variable. For more information, see "Overview of the Command Line Interface" on page 1.

To invoke cfgcheck, type the following command at the UNIX prompt:

```
$ cfgcheck -t -h -f filename.cfg
```

The **cfgcheck** command has the following arguments:

Argument	Description
No arguments (default)	When no arguments are specified and the cfgcheck utility is run by a non-root user, it searches the `cfg/ directory` for the following configuration files: initparm.cfg, netmap.cfg, userfile.cfg, and ndmapi.cfg. When a root user runs cfgcheck, the utility also searches the `SACL/ directory` to locate the sysacl.cfg file.
-h	Prints the help screen and exits.
-t	Turns on tracing and prints verbose debug information.
-f filename.cfg	Specifies a configuration file name to validate, where filename is the name of one of the configuration files. You can specify multiple -f arguments. When the -f argument is used, cfgcheck will not automatically search for other configuration files from the file specified.

Configuration Reports

You can generate a report of your system information and Sterling Connect:Direct configuration information using the Configuration Reporting Utility (cdcustrpt). Configuration reports can be generated for the following Sterling Connect:Direct components:

- Base installation of Sterling Connect:Direct

- Sterling Connect:Direct Secure Plus for UNIX
- Sterling Connect:Direct for SWIFTNet for UNIX

During the Sterling Connect:Direct installation, cdcustrpt is installed in the <*installation*>/etc/ directory.

Generating a Configuration Report on the Base Installation

Before you begin

When you use cdcustrpt to generate a report on the base Sterling Connect:Direct installation, it reports the following types of system information:

- Name and other information of the operating system
- Space on file systems
- Virtual memory statistics
- Contents of the Sterling Connect:Direct installation directory

In addition to reporting system information, cdcustrpt invokes the Configuration Checking Utility (cfgcheck) to validate the syntax of the five text-based configuration files (if they are available and if the user has access to the files) and to report on the contents of the configuration files. For more information on cfgcheck, see "Validate Configuration Files" on page 53.

In this procedure, default values are computed by the utility based on the location and name of the installed Sterling Connect:Direct and are provided in brackets "[ ]". Press **Enter** to accept the default values.

To invoke cducustrpt and generate a report of the base installation:

Procedure

1. Type the following command at a UNIX prompt:

   ```
   % cdcustrpt
   ```

2. Type the full path where Sterling Connect:Direct is installed and press **Enter**.
3. Type the full path and name for the report that will be generated and press **Enter**.

   The report is generated in the location you specified, and any error messages are displayed as shown in the following example:

   ```
   % cdcustrpt

   Enter full path of Connect:Direct destination directory:[/sci/users/jbrown1/cd40]:

   Enter full path and name for this support report file:[/sci/users/jbrown1/cd40/etc/
   cd.support.rpt]:

   ls: /sci/users/jbrown1/cd40/ndm/SACL: Permission denied

   cdcustrpt ended
   ```

   In this example, the user does not have root access, so the Strong Access Control File (sysacl.cfg) can not be accessed. The following example shows an excerpt from a sample report:

```
########################################################################
#######     Connect:Direct for UNIX 4.0.00 configuration report     #######
########################################################################
Connect:Direct for UNIX Version 4000, Build 00, IBM/RS6000 AIX, Fix date:
01OCT2007

Install directory: /sci/users/jbrown1/cd40

Local Node name: jb_aix40

Report for: jbrown1

============================================================
=====   Begin: Environment and system information   =====
============================================================

System: AIX skyglass 3 5 00CE208E4C00


Disk usage:
Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on

/dev/hd4          262144     64216   76%     2479     4% /

/dev/hd2         8126464   2708688   67%    37802     4% /usr

/dev/hd9var       262144     18448   93%      613     2% /var

/dev/hd3          786432    363600   54%      424     1% /tmp

/dev/fwdump       524288    507752    4%       17     1% /var/adm/ras/platform

/dev/hd1          262144    216520   18%      167     1% /localhome

/proc                  -         -    -         -     -  /proc

/dev/hd10opt      524288     52168   91%     3688     6% /opt

/dev/fslv00    121634816  13629040   89%   264984    15% /sci

scidalnis01:/export/nis01 1677670392 512499192   70%        0    -1% /home/nis01


Memory statistics:
System Configuration: lcpu=4 mem=3824MB

kthr    memory              page              faults        cpu
----- ----------- ------------------------ ------------ -----------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa
 1  1 400072 232777   0   0   0   0    0   0   4 1805 197  0  1 99  0
```

Generating a Configuration Report on Sterling Connect:Direct Secure Plus for UNIX

If cdcustrpt detects the Sterling Connect:Direct Secure Plus directory in the installation directory, *<installation>*/ndm/secure+/, it invokes the Sterling Connect:Direct Secure Plus Command Line Utility (splicli.sh) to report on Secure+ parameters. If Sterling Connect:Direct Secure Plus is detected, you are prompted to enter the path to the Sterling Connect:Direct Secure Plus parameters file (the default location is provided in brackets "[ ]"), for example:

```
Enter full path of Secure+ parmfile
directory: [/sci/users/jbrown1/cd40/ndm/secure+/nodes]:
```

The following example shows an excerpt from a sample report:

```
=====   Begin: Secure+ parameters   =====
========================================
All secure+ nodes:
   ************************************************************
   *               Secure+ Command Line Interface             *
   *              Connect:Direct for UNIX v4.0.00             *
   *----------------------------------------------------------*
   * Copyright (c) 1999, 2008 Sterling Commerce Inc.          *
   * All Rights Reserved.                                     *
   ************************************************************
SPCLI> display all;
  name=.Local
  baserecord=brown_aix38
  type=l
  protocol=tls
  override=n
  authtimeout=120
  stsenablesig=n
  stsenableautoupdate=n
  stslimitexportversion=y
  stsenableenc=y
  stsencalgs=(ideacbc128,tdescbc112,descbc56)
  stsauthlocalkey=0305.095A.44E3.BD87.F476.45E8.09B1.FCCA.45ED.67B0.01AD
  stsprevauthkeyexpdatetime=
  stssiglocalkey=0204.BABA.613D.2FA5.AAE6.0BD4.5847.B610.A17F.C7DD.0AA2
  stsprevsigkeyexpdatetime=
  ssltlsseaenable=n
  seacertvaldef=
  ssltlstrustedrootcertfile=/home/nis01/jbrown1/CertificateWizard/cert.crt
  ssltlscertfile=/home/nis01/jbrown1/CertificateWizard/athena.selfsigned.keycert.txt
  ssltlsenablefipsmode=n
  ssltlsenableclientauth=n
  ssltlsenablecipher=(TLS_RSA_WITH_AES_256_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA)
  2007/10/19 14:27:37 parmfile upgraded: SPV4
  2007/03/27 09:25:14 jbrown1
  2007/03/22 09:54:55 jbrown1
```

Generating a Configuration Report on Sterling Connect:Direct for SWIFTNet for UNIX

If cdcustrpt detects the SWIFTNet directory in the installation directory, <*installation*>/ndm/SwiftNet/, it includes the contents of the CDSwiftnet.cfg file in the report. Values for password parameters are replaced by a string of asterisks (*).

The following example shows an excerpt from a sample report:

```
================================================================================
Begin: /sci/users/jbrown1/swift31/ndm/SwiftNet/Version3/cfg/CDSwiftnet.cfg
================================================================================
===Content of /sci/users/jbrown1/swift31/ndm/SwiftNet/Version3/cfg/CDSwiftnet.cfg===
o
# Connect:Direct UNIX for SWIFTNet 3.1.00 configuration file.
#
[Directory.Info]
CD.HomeDir="/sci/users/jbrown1/swift31"
CDSwiftnet.HomeDir="/sci/users/jbrown1/swift31/ndm/SwiftNet/Version3"
# Concatenate the RequestorDN and ResponderDN to these directories for the Request
Handler.
Reception.Dir="/sci/users/jbrown1/reception"
Download.Dir="/sci/users/jbrown1/download"
# This directory must be specified to use the #OLDEST_FILE feature.
Success.Dir="/sci/users/jbrown1/success"
[Log.Info]
#Log.MaxSize="1048576"
Log.MaxSize="35000"
```

```
Log.MaxVersions="5"
[connection.info]
# Connection information for Connect:Direct's API port. (Used when forwarding files
to the back office.)
Comm.Info="spyglass;10102"
Userid="jbrown1"
Passwd="********  "# this is a test
#ClientInfo="/sci/users/jbrown1/swift31/ndm/SwiftNet/Version3/program/<Encrypted
userid/password file generated by the LCU>"
```

Chapter 4. Writing Custom Programs

Introduction to Writing Custom Programs

The Sterling Connect:Direct Application Programming Interface (API) allows you to write custom programs in either C or C++ to use with Sterling Connect:Direct. With the C functions or the C++ classes, you can create programs to perform the following tasks:

- Establish a connection to the Sterling Connect:Direct server
- Disconnect from the server
- Receive command responses from the server
- Send commands to the server

This topic describes the format of the Sterling Connect:Direct API functions and classes and provides samples of their use. Sample programs are provided that use the Sterling Connect:Direct API functions and classes to issue commands and receive responses from the Sterling Connect:Direct server.

Compiling Custom Programs

After you write a custom program, you must compile it, using a C or C++ compiler. Refer to the following information to determine what minimum C++ compiler version to use for each platform:

Platform	C++ Compiler
AIX	IBM XL C++ V8.0 for AIX
Sun Solaris	SPARC/x86 C++5.7
HP	aCC: HP ANSI C++ B3910B A.03.73
HP-Itanium	aCC: HP ANSI C++ B3910B A.06.07
Linux	c++ version 3.3.3

Use the commands defined in the following table to compile a custom C++ program using the C++ API calls:

Platform	C++ Compile Command
AIX	
64-bit	/usr/vacpp/bin/xlC -q64 -qinline -I../include -+ -o sdksample sdksample.C ../lib/ndmapi64.a -lbsd -ldl -lsrc -lpthreads
32-bit	/usr/vacpp/bin/xlC -qinline -I../include -+ -o sdksample sdksample.C ../lib/ndmapi.a -lbsd -ldl -lsrc -lpthreads
Sun	
32-bit	/opt/SUNWspro/bin/CC -DBSD_COMP -I../include -o sdksample sdksample.C ../lib/ndmapi.a -L/usr/ucblib -L/usr/lib -lsocket -lrpcsoc -lnsl -lelf -ldl **Note:** If /usr/ucblib is not in the LD_LIBRARY_PATH variable, add ⌂-R/usr/ucblib to the compile command.

Platform	C++ Compile Command
64-bit	/opt/SUNWspro/bin/CC -xarch=generic64 -DBSD_COMP -I../include -o sdksample sdksample.C ../lib/ndmapi64.a -L/usr/ucblib/sparcv9 -L/usr/lib/sparcv9 -L/usr/ucblib/amd64 -lsocket -lrpcsoc -lnsl -lelf -ldl -R/usr/ucblib/sparcv9 -R/usr/ucblib/amd64
HP	
32-bit	/opt/aCC/bin/aCC -AA -I../include -o sdksample sdksample.C ../lib/ndmapi.a -lrpcsoc -lnsl -ldld -Wl,+s
64-bit	/opt/aCC/bin/aCC -AA +DD64 -I../include -o sdksample sdksample.C ../lib/ndmapi64.a -L/usr/lib/pa20_64 -lnsl -ldld -Wl,+s
HP-Itanium	
32-bit	/opt/aCC/bin/aCC -I../include -o sdksample sdksample.C ../lib/ndmapi.a -lrpcsoc -lnsl -ldld -Wl,+s -lunwind
64-bit	/opt/aCC/bin/aCC +DD64 -I../include -o sdksample sdksample.C ../lib/ndmapi64.a -L/usr/lib/hpux64 -lrpcsvc -lnsl -ldld -Wl,+s -lunwind
Linux	
32-bit	g++ -m32 -I../include -O -DLINUX -o sdksample sdksample.C ../lib/ndmapi.a -ldl -lnss_nis /usr/lib/libstdc++.so.5 **Note:** If you are compiling for Linux z/OS, change -m32 to -m31.
64-bit	g++ -I../include -O -DLINUX -o sdksample sdksample.C ../lib/ndmapi64.a -ldl -lnss_nis /usr/lib64/libstdc++.so.5 **Note:** A 64-bit Linux OS installation is required to compile 64-bit binaries.

To build a C++ program using the C API calls, such as the apicheck.C sample program, replace the sdksample.C parameter with the name of the C++ program and rename the output file parameter, -o sdksample, to the name of the output file you want to create such as apicheck.

Use the commands defined in the following table to compile a C program:

Platform	C Compile Command
AIX	
32-bit	/usr/vacpp/bin/xlc -I../include -+ -o apicheck apicheck.c ../lib/ndmapi.a -lbsd -ldl -lsrc -lC -lpthreads
64-bit	/usr/vacpp/bin/xlc -q64 -I../include -+ -o apicheck apicheck.c ../lib/ndmapi64.a -lbsd -ldl -lsrc -lC -lpthreads
Sun	
32-bit	/opt/SUNWspro/bin/cc -DBSD_COMP -I../include -o apicheck apicheck.c ../lib/ndmapi.a -L/usr/ucblib -L/usr/lib -lCstd -lsocket -lrpcsoc -lnsl -lelf -ldl -lCrun **Note:** If /usr/ucblib is not in the LD_LIBRARY_PATH variable, add ⌂-R/usr/ucblib to the compile command.
64-bit	/opt/SUNWspro/bin/cc -xarch=generic64 -DBSD_COMP -I../include -o apicheck apicheck.c ../lib/ndmapi64.a -L/usr/ucblib/sparcv9 -L/usr/lib/sparcv9 -L/usr/ucblib/amd64 -lsocket -lCstd -lCrun -lrpcsoc -lnsl -lelf -ldl -lCrun -R/usr/ucblib/sparcv9 -R/usr/ucblib/amd64

Platform	C Compile Command
HP	
32-bit	/opt/ansic/bin/cc -I../include -o apicheck apicheck.c ../lib/ndmapi.a -lrpcsoc -lnsl -ldld -Wl,+s -lcl -lstd_v2 -lCsup_v2
64-bit	/opt/ansic/bin/cc +DD64 -I../include -o apicheck apicheck.c ../lib/ndmapi64.a -L/usr/lib/pa20_64 -lnsl -ldld -Wl,+s -lcl -lstd_v2 -lCsup_v2
HP-Itanium	
32-bit	/opt/ansic/bin/cc -I../include -o apicheck apicheck.c ../lib/ndmapi.a -lrpcsoc -lnsl -ldld -Wl,+s -lcl -lstd_v2 -lCsup -lunwind
64-bit	/opt/ansic/bin/cc +DD64 -I../include -o apicheck apicheck.c ../lib/ndmapi64.a -L/usr/lib/hpux64 -lrpcsvc -lnsl -ldld -Wl,+s -lcl -lstd_v2 -lCsup -lunwind
Linux	
32-bit	gcc -m32 -I../include -O -DLINUX -o apicheck apicheck.c ../lib/ndmapi.a -ldl -lnss_nis /usr/lib/libstdc++.so.5
64-bit	gcc -I../include -O -DLINUX -o apicheck apicheck.c ../lib/ndmapi64.a -ldl -lnss_nis /usr/lib64/libstdc++.so.5 **Note:** A 64-bit Linux OS installation is required to compile 64-bit binaries.
LinuxS390	
32-bit	gcc -m31 -I../include -O -DLINUX -o apicheck apicheck.c ../lib/ndmapi.a -ldl -lnss_nis /usr/lib/libstdc++.so.5
64-bit	gcc -I../include -O -DLINUX -o apicheck apicheck.c ../lib/ndmapi64.a -ldl -lnss_nis /usr/lib64/libstdc++.so.5 **Note:** A 64-bit Linux OS installation is required to compile 64-bit binaries.

Writing Custom C Programs

If you write a custom program using the C API calls, you must include the header file ndmapi.h and link it with ndmapi.a. A sample program called apicheck.c is provided.

For Java programming, you can call the C API functions by using the JNI and the libndmapi shared objects: libndmapi.sl for HP and libndmapi.so for the other supported platforms. Although the JNI is supported, the Sterling Connect:Direct Java Application Interface is recommended for Java programs that invoke the services of Sterling Connect:Direct.

Note: The environment variable NDMAPICFG must be set to the pathname of the client configuration file. Refer to "Overview of the Command Line Interface" on page 1 for instructions on setting the environment variable.

Use the following Sterling Connect:Direct API functions for C and C++ programs:

C++ Function	C Function	Description
ndmapi_connect()	ndmapi_connect_c()	Establishes a connection with the server. Specify the node to connect to in the ndm_nodespec pointer or in the CLI/API Configuration Information file. If the call is successful, NDM_NO_ERROR is returned. Control returns to the application when the connection is established and is ready for the first API request.

C++ Function	C Function	Description
ndmapi_sendcmd()	ndmapi_sendcmd_c()	Sends commands to Sterling Connect:Direct. You must provide the command text. The resp_moreflag is a pointer to the flag indicating that more responses are pending for the executed command. Invoke ndmapi_recvresp_c() for C programs or ndmapi_recvresp() for C++ programs to retrieve the extra responses. Only the select process and select statistics commands require the use of ndmapi_recvresp_c() for use with C and ndmapi_recvresp() for use with C++.
ndmapi_recvresp()	ndmapi_recvresp_c()	Receives responses to commands sent to Sterling Connect:Direct. The contents of the response buffer are returned.
ndmapi_disconnect()	ndmapi_disconnect_c()	Terminates the API connection.

Three types of Sterling Connect:Direct command responses are returned by these functions.

- Informational responses return information about the submitted command.
- Data responses, stored in the resp_buffer, contain data records.
- Error responses return ERROR_H, a pointer to a linked list of all errors found. The ID field values are fixed for use when debugging. The msgid, feedback, and rc fields are specified by Sterling Connect:Direct and are referred to in message text. The subst field points to a string that contains substitution variable information to be inserted appropriately in the message text. The error control structure keeps track of the current and total number of errors. You can move through the errors by using the next pointer in error entry blocks.

The following code defines the ERROR_H structure:

```
#define NDM_ERR_ENT_T struct NDM_ERR_ENT_S
#define NDM_ERR_ENT_H NDM_ERR_ENT_T *
#define NDM_ERR_CTL_T struct NDM_ERR_CTL_S
#define ERROR_H       NDM_ERR_CTL_T *
struct NDM_ERR_ENT_S
{
  int32    id;
  char    msgid[MSGIDLEN];
  int32    feedback;
  int32    rc;
  char    *subst;
  NDM_ERR_ENT_H     next;
};
struct NDM_ERR_CTL_S
{
  int32    id;
  int32    cur_entry;
  int32    num_entries;
  NDM_ERR_ENT_H     next;
};
```

Creating a Connection to Sterling Connect:Direct Using ndmapi_connect() or ndmapi_connect_c()

Use ndmapi_connect() or ndmapi_connect_c() to create a connection to Sterling Connect:Direct so that an application can send commands and receive responses

from the commands. Control returns to the application when the connection is established and Sterling Connect:Direct is ready for the first API request or when an error condition is set.

Following is the format for the **ndmapi_connect()** or **ndmapi_connect_c()** function:

```
int32 ndmapi_connect ERROR_H error, char *  ndm_hostname, char *  ndm_portname
```

The following table describes the parameters for the **ndmapi_connect()** or **ndmapi_connect_c()** function:

Parameter	Description	Value
error	A pointer to a Sterling Connect:Direct-defined structure that contains error information or status information.	Pointer
ndm_hostname	A pointer to the text specification of the Sterling Connect:Direct host to which the connection is made. If this parameter is not specified, the host name is determined by first checking the environment variable TCPHOST. If no value is specified, the tcp.host.name field in the CLI/API configuration file is checked. If no value is specified, the gethostbyname() command is invoked and the default value of ndmhost is used.	Null-terminated string
ndm_portname	A pointer to the host port number. If this parameter is not specified, the environment variable TCPPORT is checked. If no value is specified, the value of the tcp.port in the CLI/API configuration file is checked. If no value is specified, the default value **1363** is used.	Pointer

The **ndmapi_connect()** or **ndmapi_connect_c()** function has the following return codes:

Parameter	Description
NDM_NO_ERROR	A session was established with the server.
NDM_ERROR	A session was not established with the server. Consult the error structure for detailed error status.

The following sample function illustrates the use of **ndmapi_connect()** to connect to the sun1 host:

```
rc=ndmapi_connect (error, "sun1", "3122");
```

Terminating a Connection Using ndmapi_disconnect() or ndmapi_disconnect_c()

Use **ndmapi_disconnect()** or **ndmapi_disconnect_c()** to terminate a connection to Sterling Connect:Direct that was established by a call to **ndmapi_connect()** or **ndmapi_connect_c()**. The **ndmapi_disconnect()** or **ndmapi_disconnect_c()**function call is the following:

```
void ndmapi_disconnect
```

There are no parameters and no return codes for **ndmapi_disconnect()** or **ndmapi_disconnect_c()**. Following is a sample ndmapi_disconnect() function:

```
ndmapi_disconnect ();
```

Receiving Responses Using ndmapi_recvresp() or ndmapi_recvresp_c()

Use **ndmapi_recvresp()** or **ndmapi_recvresp_c()** to receive responses that are associated with a previous command sent from the application. Following is the format for **ndmapi_recvresp()** or **ndmapi_recvresp_c()**:

```
int32 ndmapi_recvresp ERROR_H  error int32 *  resp_length, char *   resp_buffer,
int32 * resp_moreflag
```

Following are the parameters for ndmapi_recvresp() or **ndmapi_recvresp_c()**:

Parameter	Description	Value
error	A pointer to a Sterling Connect:Direct-defined structure that contains error information or status information.	Pointer
resp_length	A pointer to the length, in bytes, of the application buffer to receive the response. The API sets this parameter to the number of bytes returned.	Pointer to number of bytes returned or 0 if you no longer want to receive responses. Setting this field to zero purges all queued responses.
resp_buffer	A pointer to the application buffer that receives the command or submit response. This buffer should allocate 4096 bytes. The format of **resp_buffer** is a free-form text record structure. Field names are four characters long and all uppercase. The data can be any length and can include blanks. The structure is: field name=data | field name=data |... For example: SUBM = username | PNUM = 12 | PNAM = proc1 |...	A local buffer, with a size greater than or equal to that set by **r**esp_length and filled in by **ndmapi_recvresp()** or **ndmapi_recvresp_c()**. The CLI passes the resp_buffer to AWK for parsing. Valid values include: ADMN—Sterling Connect:Direct administrator name ADPH—Sterling Connect:Direct administrator phone number CCOD—Completion code CKPT—Checkpoint CLAS—Class DBYW—Bytes written DBYX—Bytes received DCOD—Destination completion code DDAY—Submit date

Parameter	Description	Value
		DDS1—Destination disposition 1 DDS2—Destination disposition 2 DDS3—Destination disposition 3 DESC—Sterling Connect:Direct administrator description DFIL—Destination file DMSG—Destination message ID DNVL—Destination number of volumes DRCW—Records written DRUX—RUs received DVOL—Destination volume array ECMP—Extended compression ON or OFFFROM—Copy sending node LCCD—Local completion code LCLP—Local IP address and port number LKFL—Link fail LMSG—Local message ID LNOD—Local node MSGI—Message IDMSGT—Message text MSST—Short text OCCD—Other completion code OERR—Other node in error OMSG—Other message ID PACC—PNODE account PFIL—Process file PNAM—Process name

Parameter	Description	Value
		PNOD—PNODE PNUM—Process number PPMN—PDS member name PRTY—Priority QUEU—Queue RECC—Record category RECI—Record ID RETA—Retain Process RMTP—Remote IP address and port number RSTR—Process restarted RUSZ—RU Size SACC—SNODE account SBID—Submitter node ID SBND—Submitter node name SBYR—Bytes read SBYX—Bytes sent SCMP—Standard compression SCOD—Source completion code SDDY—Schedule date SDS1—Source disposition 1 SDS2—Source disposition 1 SDS2—Source disposition 2 SDS3—Source disposition 3SELA—Elapsed time of the event SFIL—Source file SMSG—Source message ID SNAM—Step name SSTA—Start time of the event STAR—Start log date/time for record

Parameter	Description	Value
		STAT—Process status SNOD—SNODE SNVL—Source number of volumes SOPT—SYSOPTS record SRCR—Records read SRUX—RUs sent STIM—Schedule time STOP—Stop time of the event SUBM—Submitter ID SUBN—Submitter node SUMM—Summary output selector SVOL—Source volume array TIME—Submit time XLAT—Translation
resp_moreflag	Indicates that more ndmapi_recvresp() or ndmapi_recvresp_c() calls must be issued for more information. This flag occurs only on select process and select statistics commands.	None

The **ndmapi_recvresp()** or **ndmapi_recvresp_c()** function has the folliowing return codes:

Return Code	Description
NDM_NO_ERROR	The function completed successfully.
NDM_ERROR	An error occurred. Consult the error structure for detailed error status.
TRUNCATED	Data is truncated because the receiving buffer is too small.

Following is a sample **ndmapi_recvresp()** function:

```
int32 rc, resp_length;
int32 resp_moreflag;
char resp_buffer[makbuf];

rc= ndmapi_recvresp (error,
                     &resp_length,
         resp_buffer,
         &resp_moreflag
         );
```

Sending a Command to Sterling Connect:Direct Using ndmapi_sendcmd() or ndmapi_sendcmd_c()

Use **ndmapi_sendcmd()** or **ndmapi_sendcmd_c()** to allow a command to be sent to a Sterling Connect:Direct application. Following is the format of ndmapi_sendcmd() or **ndmapi_sendcmd_c()**:

```
int32  rc, resp_moreflag;
struct sendcmd_data ret_data;
rc=ndmapi_sendcmd (error,
                   "select process pnumber=2,",
                   &resp_moreflag,
         &ret_data
        );
```

Following are the parameters for **ndmapi_sendcmd()** or **ndmapi_sendcmd_c()**:

Parameter	Description	Value
error	A pointer to a Sterling Connect:Direct-defined structure that contains error information or status information.	Pointer
cmd_text	A pointer to the null-terminated text string that specifies the command to send to Sterling Connect:Direct. The command text must be followed by a semicolon and terminated with a null. When you use the **submit=filename** command from the API, ensure that you allocate enough storage for the Process text. The text of the Process submitted is returned in the text string associated with this parameter when the function completes. If you do not allocate enough storage for the Process text, a core dump can result.	Pointer to a text string
resp_moreflag	A pointer to the flag that indicates that more responses are pending for the command just executed. Invoke **ndmapi_recvresp()** or **ndmapi_recvresp_c()** to retrieve the extra responses.	Pointer to a flag

ret_data	A pointer to a structure containing internal response information for a command. The structure is:
	`struct sendcmd_data  {`
	`  char  * cmd_name;`
	`  ulong cmd_id;`
	`  long data1;`
	`  long data2;`
	`  long data3;`
	`};`
		Pointer to a structure

sendcmd_data	Provides the caller with some information about the user request. Because parsing of command text occurs at the CMGR, the End User Application (EUA) has no way to identify the command that was submitted, unless it generated the text.	Information about the user request
cmd_name	A pointer to a string with the name of the command submitted. The CLI uses this pointer to display completion messages. This field enables you to display unique completion messages without any knowledge of a specific command in the EUA.	Pointer to name of command

Parameter	Description	Value

cmd_id	A four-byte identifier of the command that was found in the command text. Following are the four-byte identifiers:
	`/*************Command IDs******************/`
	`#define CHANGE_PROCESS 0x43484750 /* "CHGP" */`
	`#define DELETE_PROCESS 0x44454c50 /* "DELP"*/`
	`#define FLUSH_PROCESS 0x464c5350 /* "FLSP" */`
	`#define SELECT_PROCESS 0x53454c50 /* "SELP"*/`
	`#define SELECT_STATISTICS 0x53454c53 /* "SELS" */`
	`#define SUBMIT 0x5355424d /* "SUBM" */`
	`#define TRACE_API 0x41504920 /* "API " */`
	`#define TRACE_CMGR 0x434d4752 /* "CMGR" */`
	`#define TRACE_SMGR 0x534d4752 /* "SMGR" */`
	`#define TRACE_PMGR 0x504d4752 /* "PMGR" */`
	`#define TRACE_COM 0x434f4d4d /* "COMM"*/`
	`#define TRACE 0x54524143 /* "TRAC" */`
	`#define STOPNDM 0x53544F50 /* "STOP" */`
	
	The CLI uses these identifiers to ensure that rules are being followed. For instance, if an **ndmapi_sendcmd** returns with the **resp_moreflag** set and the **cmd_id** is not **SELECT_STATISTICS** or **SELECT_PROCESS**, the CLI generates an error.
		Four-byte identifier

data1, data2, and data3	For future expansion. data1 is used with the **submit** command to return the Process number. data2 is used with the submit command to return the result of the Process (0, 4, 8, or 16)	

The **ndmapi_sendcmd_c()** function call has the following return codes:

Return Code	Description
NDM_NO_ERROR or Process Number	The function completed successfully.
NDM_ERROR	An error occurred. Consult the error structure for detailed error status.

Following is a sample ndmapi_sendcmd() function:

```
int32  rc, resp_moreflag;
struct sendcmd_data ret_data;
rc=ndmapi_sendcmd (error,
                   "select process pnumber=2 ;",
                   &resp_moreflag,
         &ret_data
         );
```

Writing Custom C++ Programs

If you write a custom program using C++ API calls, you must include the class called ConnectDirectSession. The calling program must instantiate ConnectDirectSession and call the send and receive functions. A sample program called sdksample.C is provided. To write a custom C++ program, create a ConnectDirectSession class. The class contains the ConnectDirectSession interface and a constructor and destructor call to allocate and release the storage associated with the class. This class is the interface to the Sterling Connect:Direct methods and provides connection, command, data retrieval, and error services. Each method returns either CD_SUCCESS or CD_FAILURE.

Note: The environment variable NDMAPICFG must be set to the pathname of the client configuration file. Refer to "Starting the CLI" on page 1for instructions on setting the environment variable.

To use the ConnectDirectSession class, your application must include the cdunxsdk.h header file provided in the installation and must link with the ndmapi.a file. Following is a sample ConnectDirectSession class program:

```
#include "cdunxsdk.h"
#include <iostream.h>
#include <string.h>
void getError(ConnectDirectSession& cdSess);
main()
{
    ConnectDirectSession cdSess;
    char processText[16384];
    if (cdSess.SessionINF->Connect() == CD_SUCCESS)
    {
        strcpy(processText,"submit maxdelay=unlimited sdksample process snode=SNODENAME ");
        strcat(processText,"copy00 copy from (file=sample.txt pnode)");
        strcat(processText,"              to (file=sample.000 snode disp=rpl) ;");
        if (cdSess.SessionINF->SendCommand(processText) == CD_SUCCESS)
        {
         printf("%s completed, pnumber = %ld.\n",
           cdSess.SessionINF->GetCommandName(),
           cdSess.SessionINF->GetProcessNumber());
sprintf(processText, "SELECT STATISTICS PNUMBER=%ld DETAIL=YES ;", cdSess.SessionINF-
>GetProcessNumber());
(cdSess.SessionINF->SendCommand(processText) == CD_SUCCESS)
            {
 }
     else
          {
         getError(cdSess);
     }
   }
        else
        {
     getError(cdSess);
   }
 cdSess.SessionINF->DisConnect();
    }
    else
    {
        getError(cdSess);
    }
}
void getError(ConnectDirectSession& cdSess)
{
    if (cdSess.SessionINF->GetFirstError())
    {
        printf("\nError Message:  %s",   cdSess.SessionINF->GetMsgID());
        printf("\nError Feedback: %d",   cdSess.SessionINF->GetFeedBackCode());
        printf("\nError RC:       %d",   cdSess.SessionINF->GetReturnCode());
        printf("\nError SUBST:    %s\n", cdSess.SessionINF->GetSubstitute());    }
 while(cdSess.SessionINF->GetNextError())
    {
        printf("\nError Message:  %s",   cdSess.SessionINF->GetMsgID());
        printf("\nError Feedback: %d",   cdSess.SessionINF->GetFeedBackCode());
        printf("\nError RC:       %d",   cdSess.SessionINF->GetReturnCode());
        printf("\nError SUBST:    %s\n", cdSess.SessionINF->GetSubstitute());
        }
}
```

The ConnectDirectSession class methods are described in the following table:

Method	Description	Parameter	Return Values
Connect	Provides a connection to the Sterling Connect:Direct server. Connect() with a void parameter connects to the hostname and port specified in the client configuration file.	void or a pointer to an IP address and port.	CD_SUCCESS or CD_FAILURE
DisConnect	Disconnects the current session.	void	CD_SUCCESS or CD_FAILURE
SendCommand	Sends a Sterling Connect:Direct command to the server for processing.	Pointer to a command text buffer.	CD_SUCCESS or CD_FAILURE
ReceiveResponse	Receives the response from a previously issued command, such as the select statistics command.	void	CD_SUCCESS or CD_FAILURE
GetResponse	Retrieves the response from the ReceiveResponse call.	void	Pointer to a response buffer.
GetResponseLength	Returns the length of the previously received response buffer.	void	Length of the response buffer from the previously issued call.
MoreData	Returns a value indicating if outstanding data from the previously issued send command call is available. If the return value is TRUE, call ReceiveResponse again to retrieve more data.	void	TRUE—If more data is outstanding. FALSE—If no data is outstanding.
GetCommandName	Returns the command name of the previously issued send command, such as the submit command.	void	Pointer to a command name buffer.
GetProcessNumber	Returns the Process number of a previously issued submit command.	void	Process number of a submit command. -1—If no submit command can be found.
GetProcessCount	Returns the number of Processes affected by the last send command that issued a delete, change, or flush process.	void	Process number of a submit command that issued a delete, change or flush process. -1—If no submit command can be found.
GetCurrentError	Moves the error data pointer to the current error in the list.	void	TRUE—If successful FALSE—If no current error exists.
GetNextError	Moves the error data pointer to the next error in the list.	void	TRUE—If successful FALSE—If no more errors are found.

Method	Description	Parameter	Return Values
GetPreviousError	Moves the error data pointer to the previous error in the list.	void	TRUE—If successful FALSE—If no previous error exists.
GetFirstError	Moves the error data pointer to the first error in the list.	void	TRUE—If successful FALSE—If no error is found.
GetLastError	Moves the error data pointer to the last error in the list.	void	TRUE—If successful, otherwise FALSE.
GetMsgID	Retrieves the message of the current error data block. You must call one of the GetXXXXError methods before calling this method in order to retrieve the proper results.	void	Return Value: Pointer to a message ID if data block is value.
GetFeedBackCode	Returns the feedback code of the current error data block.	void	Feedback code.
GetReturnCode	Returns the Sterling Connect:Direct return code.	void	One of the valid Sterling Connect:Direct return code: 1,4,8,16.
GetStatus	Returns the status.	void	Sterling Connect:Direct status code.
GetSubstitute	Returns the current substitution buffer associated with the error.	void	Pointer to a substitution buffer.
DisplayError	Displays the current error chain to an output location.	Parameters: Pointer to a file I/O structure.	Return Value: Returns the highest error found in the error chain or -1 on error.

Following is the ConnectDirectSession class header:

```
#include <stdio.h>

// Error enumeration.
typedef enum CDErrorCode
{
    CD_SUCCESS =  0,
    CD_FAILURE = -1

} CDErrorCode;

// <<Interface>>
class CDSession
{
public:
    // Communication methods...
    virtual CDErrorCode Connect(void) = 0;
    virtual CDErrorCode Connect(char *IpAddress, char *IpPort) = 0;
    virtual CDErrorCode DisConnect(void) = 0;
    virtual CDErrorCode SendCommand(char *CmdText) = 0;
    virtual CDErrorCode ReceiveResponse(void) = 0;
```

```
    // Methods for retrieving ReceiveResponse data...
    virtual const char *GetResponse(void) = 0;
    virtual int         GetResponseLength(void) = 0;
    virtual bool        MoreData(void) = 0;

    // Methods for retrieving SendCommand return data...
    virtual const char *GetCommandName(void) = 0;
    virtual long        GetProcessNumber(void) = 0;
    virtual long        GetProcessCount(void) = 0;

// Methods to iterate over error collection ...
    virtual bool        GetCurrentError(void) = 0;
    virtual bool        GetNextError(void) = 0;
    virtual bool        GetPreviousError(void) = 0;
    virtual bool        GetFirstError(void) = 0;
    virtual bool        GetLastError(void) = 0;

    // Methods to retrieve error data...
    virtual const char *GetMsgID(void) = 0;
    virtual int         GetFeedBackCode(void) = 0;
    virtual int         GetReturnCode(void) = 0;
    virtual int         GetStatus(void) = 0;
    virtual const char *GetSubstitute(void) = 0;

    // Method to display error collection...
    virtual int   DisplayError(FILE *Output) = 0;
};

class ConnectDirectSession
{
public:
    // Interface classes
    CDSession *SessionINF;

    ConnectDirectSession();
    ~ConnectDirectSession();
};
```

Chapter 5. Writing User Exits

User Exit Programs

The user exit API functions allow you to write custom programs to use with Sterling Connect:Direct. The user exit programs are used by Sterling Connect:Direct to invoke user-specific logic at strategic points within Sterling Connect:Direct execution. User exit programs must be C or C++ language programs and cannot be shell scripts. The PMGR invokes the Statistics user exit program when you start Sterling Connect:Direct and the exit runs as long as Sterling Connect:Direct runs. The SMGR invokes the File Open and Security user exits for each session and stops them when the particular session terminates.

Note: exit_skeleton.c and exit_skeleton.C contain working examples of all three exits and can be made with the make_exit_c and make_exit_C make files.

The user exit programs are described in the following:

Program	Description

File Open Exit	Sterling Connect:Direct sends a message to this user exit program to open the source or destination file during processing of the copy statement. The File Open Exit opens the source file and identifies the file descriptor. This exit can perform any sort of processing to file names or directory names. It can also redirect the open request to other files as needed.
	
	The File Open Exit program (named "exit_skeleton" in this example) must be owned by root and the setuid bit must be set. Use the following commands:
	
	`% chown root exit_skeleton`
	
	`% chmod u+s exit_skeleton`

Security Exit	The Security Exit enables you to implement your own security system or provide access to a third-party security system.
Statistics Exit	The Security Exit enables you to implement your own security system or provide access to a third-party security system.

User Exit Functions

A connection between the user exit and Sterling Connect:Direct is established when the user exit program calls the **exit_child_init()** or **exit_child_init_c()** function. The connection is terminated through a specially designated stop message. The types of messages are defined in the include file user_exit.h. The following functions facilitate communications between the user exit and Sterling Connect:Direct:

C++ Function	C Function	Description
exit_child_init()	exit_child_init_c()	Use this function as the first line in a user exit program to initialize communications between Sterling Connect:Direct and the user exit program.
recv_exit_msg()	recv_exit_msg_c()	Used by both Sterling Connect:Direct and the user exit program to receive a message from the other Process. The receive exit messages wait for a response from the other Process.
send_exit_file()	send_exit_file_c()	The user exit program uses this function when it has opened a file for Sterling Connect:Direct. This function uses underlying UNIX methods to pass an open file descriptor. from one Process to another.
send_exit_msg()	send_exit_msg_c()	Both Sterling Connect:Direct and the user exit program use this function to send a message to the other Process. Send messages are followed with a receive message to get the response from the other Process.

Initializing Communications with exit_child_init() or exit_child_init_c()

Use the **exit_child_init()** or **exit_child_init_c()** function as the first line of code of the user exit program to initialize communications. This function performs a check to verify that each side is ready to communicate. Following is the format of the **exit_child_init()** function:

```
int exit_child_init( char * logfile )
```

The exit_child_init() or exit_child_init_c() function has the following parameter:

Parameter	Description	Value
logfile	The name of the log or trace file that is opened for use by the user exit programs. Because the file open and security exit are started by SMGR, which is running as root, the exits also run as root. Running the exits as root can cause problems with file permissions of the log file, so **logfile** enables you to easily change owner or permissions on the file. See the sample exit in *d_dir*/ndm/src/exit_skeleton.c for more details.	Name of log file or trace file

The **exit_child_init()** or **exit_child_init_c()** function have the following return codes. Return codes for the function are defined in ndmapi.h.

Return Code	Description
GOOD_RC	Communications between Sterling Connect:Direct and the user exit program were successfully initialized.
ERROR_RC	Communications between Sterling Connect:Direct and the user exit program could not be initialized.

Waiting for a Message Using recv_exit_msg() or recv_exit_msg_c()

The recv_exit_msg() or recv_exit_msg_c() function waits until it receives a message from Sterling Connect:Direct. Control is suspended until a message is received or an error occurs. The recv_exit_msg() has the following format:

```
int recv_exit_msg( int exit_flag )
 int * msg_type,
 char * recv_buf,
 int * recv_buf_len
```

The recv_exit_msg() or recv_exit_msg_c() functions have the following parameters:

Parameter	Description	Value
exit_flag	A flag to specify the recipient ID. The only valid value a user exit program can use is EXIT_PROGRAM.	EXIT_PROGRAM
msg_type	A pointer to the name of the received message. Messages are requests from Sterling Connect:Direct and the associated response from the user exit program.	Pointer to message
recv_buf	A pointer to the memory location of the message.	Pointer to message
recv_buf_len	The length in bytes of the message to be received.	Length of message

The recv_exit_msg()or recv_exit_msg_c() functions have the following return codes. Return codes for the function are defined in ndmapi.h.

Return Code	Description
GOOD_RC	The message was received successfully.
ERROR_RC	An error occurred and the message was not received successfully. Possible causes include: Sterling Connect:Direct terminated, an invalid value used for the exit_flag parameter, or the receiving buffer not large enough to hold the message received.

Passing a File Descriptor Using send_exit_file() or send_exit_file_c()

Use the send_exit_file() or send_exit_file_c() function to pass a file descriptor from one Process to another Process. Following is the format of send_exit_file():

```
int send_exit_file int  exit_flag
 int fd
```

Following are the parameters for send_exit_file() or send_exit_file_c():

Parameter	Description	Value
exit_flag	A flag to specify the sender ID. The only valid value a user exit program can use is EXIT_PROGRAM.	EXIT_PROGRAM
fd	The file descriptor of a file that the user exit program opened in the place of Sterling Connect:Direct, similar to one returned by the open(2) function.	File descriptor

The send_exit_file() or send_exit_file_c() function calls have the following return codes. Return codes for the function are defined in ndmapi.h.

Header	Header
GOOD_RC	The file descriptor was received successfully.
ERROR_RC	An error occurred and the file descriptor was not sent successfully. Possible causes include: Sterling Connect:Direct terminated, an invalid value used for the exit_flag or fd parameters, or the last message sent was not send_exit_msg.

Sending a Message to Sterling Connect:Direct Using send_exit_msg() or send_exit_msg_c()

The send_exit_msg() or send_exit_msgc() function enables the user exit program to send a message to Sterling Connect:Direct. This function returns control to the caller immediately after the message is queued.

Following is the format of the send_exit_msg() function:

```
int send_exit_msg int exit_flag
 int msg_type,
 char * send_buf,
 int send_buf_len
```

Following are the parameters for send_exit_msg() or send_exit_msg_c():

Parameter	Description	Value
exit_flag	A flag to specify the sender ID. The only valid value a user exit program can use is EXIT_PROGRAM.	EXIT_PROGRAM
msg_type	A message name. Messages are requests from Sterling Connect:Direct and the associated response from the user exit program.	Pointer to message
send_buf	A pointer to the memory location of the message to be sent.	Pointer to message
send_buf_len	The length in bytes of the message to be sent.	Length of message

Following are the return codes for **send_exit_msg()** or **send_exit_msg_c()**. Return codes for the function are defined in ndmapi.h.

Return Code	Description
GOOD_RC	The message was sent successfully.
ERROR_RC	An error occurred and the message was not sent successfully. Possible causes include: Sterling Connect:Direct terminated or an invalid value is used for the exit_flag or msg_type parameters.

Overview of User Exit Messages

Sterling Connect:Direct sends and receives messages, using the send_exit_msg() and the **recv_exit_msg()** functions for a C++ program or the **send_exit_msg_c()** and the **recv_exit_msg_c()** functions for a C program. For the exact definition of the data sent in each message, see the files located in *d_dir*/ndm/include/user_exit.h and *d_dir*/ndm/include/user_exit2.h.

Note: The copy control block is defined in user_exit2.h.

Statistics Exit Message

The statistics exit has only one type of message, the STATISTICS_LOG_MSG.

Sterling Connect:Direct sends a STATISTICS_LOG_MSG to the user exit program. Every time Sterling Connect:Direct writes a statistic record, this message provides an exact copy of the character string. The STATISTICS_LOG_MSG contains the Sterling Connect:Direct statistics record.

File Open Exit Messages

The file open exit has four types of messages:

- FILE_OPEN_OUTPUT_MSG
- FILE_OPEN_OUTPUT_REPLY_MSG
- FILE_OPEN_INPUT_MSG
- FILE_OPEN_INPUT_REPLY_MSG

The file open exit has the following limitations:

- The oflag parameter passed to the user exit is already calculated based on the file disposition, as explicitly specified on the copy statement or using the default value. If the user exit changes the oflag to truncate and the original disposition is mod meaning the copy will append to the end of file if the file already exists, then the user exit causes the Process to behave differently from how the Process language is documented.
- Do not change the file type specified by the Process. For example, if the Process specifies a regular file, the user exit cannot open and return a file descriptor for a pipe. No facility is available to modify contents of the copy control block and return it to Sterling Connect:Direct.
- If the oflag specifies opening a file with write access and the user exit changes access to read-only, Sterling Connect:Direct will fail when it attempts to write to a read-only file.
- The upload and download parameters that restrict directory access are ignored for this user exit.

FILE_OPEN_OUTPUT_MSG

During the copy statement process, Sterling Connect:Direct sends a FILE_OPEN_OUTPUT_MSG to the user exit program to open the destination file. The FILE_OPEN_OUTPUT_MSG contains:

- The open function oflag parameter (for example, O_CREAT|O_RDWR|O_TRUNC)
- The open function mode parameter, which controls file permissions
- UNIX user ID that will own the file
- UNIX group ID that will own the file

- UNIX user name
- A copy of the Sterling Connect:Direct copy control block
- A copy of the Sterling Connect:Direct parsed sysopts structure (the copy control block contains the actual raw version from the process)

FILE_OPEN_OUTPUT_REPLY_MSG

The user exit program sends a reply message to the Sterling Connect:Direct FILE_OPEN_OUTPUT_MSG. The FILE_OPEN_OUTPUT_REPLY_MSG contains:

- Status value of zero for successful or non zero for failure
- Status text message (if status value is failure, status text message is included in the error message)
- Pipe pid (for pipe I/O, the UNIX process ID of the shell process that is performing the shell command for pipe I/O)
- Actual file name opened (to be used in statistics log messages)

If the status value is zero for successful, the user exit program must immediately call **send_exit_file()** or **send_exit_file_c()** to send the file descriptor of the opened file to Sterling Connect:Direct.

FILE_OPEN_INPUT_MSG

During the copy statement Process, Sterling Connect:Direct sends a FILE_OPEN_INPUT_MSG to the user exit program to open the source file. The FILE_OPEN_INPUT_MSG contains:

- The open function oflag parameter (for example, O_RDONLY)
- The open function mode parameter, which controls file permissions
- UNIX user ID that will own the file
- UNIX group ID that will own the file
- UNIX user name
- A copy of the Sterling Connect:Direct copy control block
- A copy of the Sterling Connect:Direct parsed sysopts structure (the copy control block contains the actual raw version from the Process)

FILE_OPEN_INPUT_REPLY_MSG

This message type is used when the user exit program sends a reply message to the Sterling Connect:Direct FILE_OPEN_INPUT_MSG. The FILE_OPEN_INPUT_REPLY_MSG contains:

- Status value of zero for success or non zero for failure
- Status text message (if status value is failure, status text message is included in the error message)
- Pipe pid (for pipe I/O, the UNIX process ID of the shell process that is performing the shell command for pipe I/O)
- Actual file name opened (used in statistics log messages)

Security Exit Messages

The security exit contains four types of messages:

- GENERATE_MSG
- GENERATE_REPLY_MSG

- VALIDATE_MSG
- VALIDATE_REPLY_MSG

CAUTION:
If the security exit is used, Sterling Connect:Direct relies on it for user ID authentication. If the security exit is not implemented correctly, security can be compromised.

GENERATE_MSG

Sterling Connect:Direct sends a generate message to the user exit program at the start of a session to establish a security environment. The PNODE sends the GENERATE_MSG to the security exit to determine a user ID and security token to use for authentication on the SNODE. The GENERATE_MSG contains:

- Submitter ID
- PNODE ID
- PNODE ID password, if user specified one
- SNODE ID
- SNODE ID password, if user specified one
- PNODE name
- SNODE name

GENERATE_REPLY_MSG

The user exit program sends a reply message to Sterling Connect:Direct. The GENERATE_REPLY_MSG contains:

- Status value of zero for success or non zero for failure
- Status text message (if status value is failure, status text message is included in the error message)
- ID to use for security context on the SNODE side (may or may not be the same ID as in the generate message)
- Security token used in conjunction with ID for security context on the SNODE side

VALIDATE_MSG

Sterling Connect:Direct sends a validate message to the user exit program. The SNODE sends the VALIDATE_MSG to the security exit to validate the user ID and security token received from the PNODE. The VALIDATE_MSG contains:

- Submitter ID
- PNODE ID
- PNODE ID password, if user specified one
- SNODE ID
- SNODE ID password, if user specified one
- PNODE name
- SNODE name
- ID to use with security token
- Security token (password, PASSTICKET, or other security token)

VALIDATE_REPLY_MSG

The user exit program sends a reply message to the Sterling Connect:Direct VALIDATE_MSG. The VALIDATE_REPLY_MSG contains:

- Status value of zero for success or non zero for failure
- Status text message (if status value is failure, status text message is included in the error message)
- ID used for security context
- Security token to use in conjunction with ID for security context

User Exit Stop Message

Sterling Connect:Direct sends the stop message, STOP_MSG, when all useful work for the user exit is complete and to notify the user exit to terminate. A user exit should terminate only when a stop message is received or if one of the above listed user exit functions returns an error code.

Copy Control Block

The copy control block structure contains the fields, which control how Sterling Connect:Direct Processes the copy statement Process file.

Exit Log Files

If user exit programs are specified in the initparm.cfg, Sterling Connect:Direct creates exit logs. Exit log files are provided specifically for the user exit programs and are used for debug and trace type messages. The user exit program is started with the log file already opened on STDOUT and STDERR. The exit log files are:

- stat_exit.log
- file_exit.log
- security_exit.log

Note: You can access the log files through the normal **printf()** and **fprintf (stderr,...)** functions.

The log files are located in the installed (d_dir) working directory:

.../*d_dir*/work/cd_node

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing

Legal and Intellectual Property Law

IBM Japan Ltd.

19-21, Nihonbashi-Hakozakicho, Chuo-ku

Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be

incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

J46A/G4

555 Bailey Avenue

San Jose, CA 95141-1003

U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© IBM 2013. Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2013.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium and the Ultrium Logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Connect Control Center®, Connect:Direct®, Connect:Enterprise®, Gentran®, Gentran®:Basic®, Gentran:Control®, Gentran:Director®, Gentran:Plus®, Gentran:Realtime®, Gentran:Server®, Gentran:Viewpoint®, Sterling Commerce™, Sterling Information Broker®, and Sterling Integrator® are trademarks or registered trademarks of Sterling Commerce®, Inc., an IBM Company.

Other company, product, and service names may be trademarks or service marks of others.

Index

Special characters

A

C

D

E

F

G

H

V

W

Product Number: 5725-C99

Printed in USA