Sterling Connect:Direct

Overview

March 31, 2012

Sterling Connect:Direct

Overview

March 31, 2012

Note
Before using this information and the product it supports, read the information in "Notices" on page 45.

This edition applies to the March 31, 2012 version of IBM Sterling Connect:Direct Process Language Reference Guide and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. What is a Sterling Connect:Direct Process?

What is a Sterling Connect:Direct Process?

An IBM® Sterling Connect:Direct® Process is a series of statements and parameters that perform data movement and manipulation activities such as:

- moving files between different Sterling Connect:Direct servers
- running jobs, programs, and commands on the Sterling Connect:Direct server
- starting other Processes
- monitoring and controlling Processes
- handling processing errors

Processes can be linked to network and application activities, generating a continuous cycle of processing. For example, a network message can trigger a file transfer that is used by another application.

As a Process executes and after it completes, audit information is available to analyze and use for future processing.

Features that you can specify in a Process include:

- Scheduling–Setting a Process to run at a specific day and time. Processing can be scheduled to run automatically at a specified date or interval, without any operator intervention.
- Integration with native applications–Invoke native applications from within a Sterling Connect:Direct Process.
- Integration with existing security systems–Specify user IDs and passwords within a Process that allow it to work within your existing network security system
- Data transmission integrity–Specifying checkpoint and restart intervals within a file transmission, so that if a transmission fails at some point, it can be restarted automatically from the most recent checkpoint.
- User notification–Automatically notify users of successful and unsuccessful transfers.

Process Language

A Sterling Connect:Direct Process uses its own scripting language that defines the work that you want the Process to do. The following are the statements used in Sterling Connect:Direct Processes:

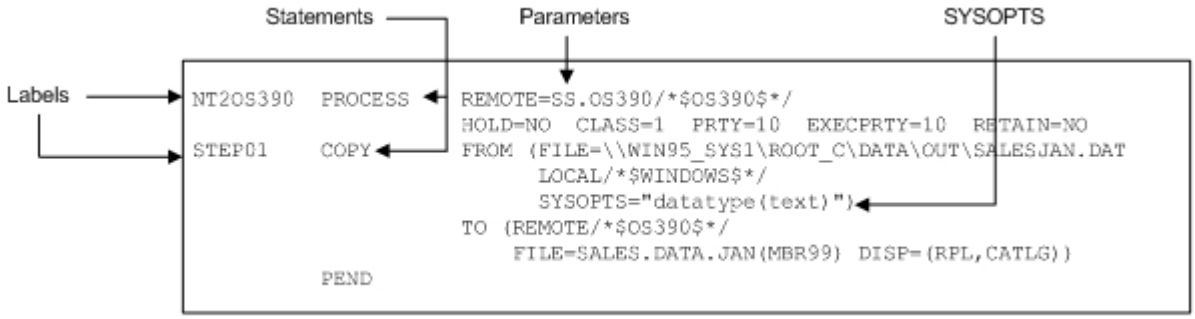
Statement	Description
PROCESS	Defines general Process characteristics. This statement is always the first statement in a Process. Among the items the Process statement specifies are: • The name of the secondary node in the Process • The Process priority • When to start the Process • Who to notify upon completion • Whether Sterling Connect:Direct should keep a copy of the Process to execute in the future
COPY	performs a data transfer. The COPY statement also specifies various file transfer options, including: File allocation File disposition options File renaming Data compression options
RUN JOB	submits a job or application to the host operating system. The Process continues running and does not wait for the submitted job or application to complete. This is known as asynchronous processing.
RUN TASK	submits a job or application to the host operating system. The Process waits for the job or application to complete before continuing. If the job or application does not complete, the rest of the Process does not run. This is known as synchronous processing.
SUBMIT	submits a Process from within another Process. The SYMBOL statement enables Processes to be modular. This enhances processing flexibility, as you can modify Process modules as necessary without altering the master Process.
SYMBOL	replaces symbolic strings within a Process with parameter values. The SYMBOL statement eliminates the need to hard-code file names and values within a Process. Instead, the SYMBOL statement allows values to be substituted within a Process, enabling a Process to be reused for different file transfers.
Conditional (IF, EIF, ELSE, EXIT, GOTO	controls Process execution by testing Process step return codes with conditional logic statements. For example, if a file transfer successfully completes, the Process can use the SUBMIT statement to initiate a second Process. If the Process transfers files, it can also send an error message to the operator.
pend	indicates the end of a Process. This statement is only valid for Sterling Connect:Direct for UNIX and Microsoft Windows.

The Process statement must be the first statement in a Process. The statements after the Process statement can follow in any sequence. Each statement uses parameters to control Process activities such as execution start time, user notification, security,

or accounting data. These parameters can be specified within the Process or you can specify them when you submit the Process. The parameters for a statement vary according to platform.

Process Components

The following example shows the components of a Sterling Connect:Direct Process:



Note: The maximum size allowed for a Process is 1 MB.

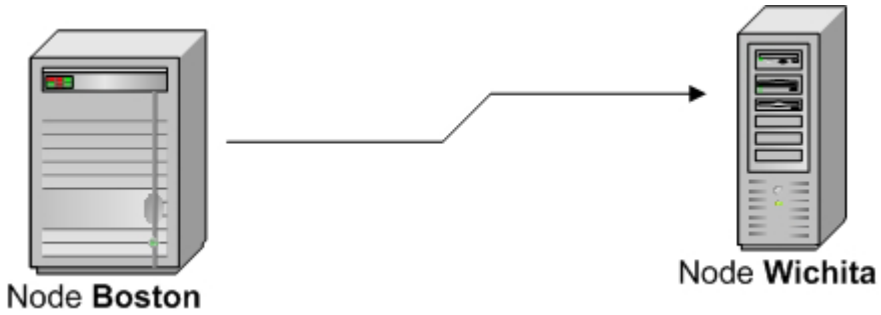
The following table describes these components:

Component	Description
Label	A user-defined 1-8 character alphanumeric string that identifies the Sterling Connect:Direct statement. The label must begin in column one. The first character of the label must be alphabetic. The PROCESS statement is the only statement that requires a label; it is optional on all other statements. The PROCESS statement must be on the same line as the label.
Statement	The statement specifies the requested function Statements must begin after column one. If a statement begins in column one, Sterling Connect:Direct considers it a label. For more information on how to code Sterling Connect:Direct statements to transfer files and submit business processes to IBM Sterling B2B Integrator, see Transmitting Files between Sterling Connect:Direct and Sterling B2B Integrator.

Component	Description
Parameters and Subparameters	Parameters and subparameters specify further instructions for the statement. They must be separated from statements by one or more blanks or commas. Multiple symbolic substitutions must be separated by one or more spaces. There are two types of parameters: keyword and positional. • Keyword–Keyword parameters are usually followed by an equal sign and may have a set of subparameters. • Positional–Positional parameters must be entered in a specific order, with commas replacing any parameter omitted. These parameters are always to the right of the equal sign. Positional parameters must be enclosed in parentheses, with the parentheses optionally preceded and followed by blanks or commas. A positional parameter or the variable information in a keyword parameter sometimes is a list of subparameters. The list may contain both positional and keyword parameters. Positional subparameters must be enclosed in parentheses, with the parentheses optionally preceded and followed by blanks or commas.
SYSOPTS	SYSOPTS (system operations) are a specialized type of parameter used by every Sterling Connect:Direct platform. SYSOPTS specify platform-specific commands to perform during a Process. For example, when transferring a file from a mainframe system to a UNIX system, you use SYSOPTS to specify that the file be translated from EBCDIC to ASCII and that any trailing blanks be removed. See SYSOPTS Syntax for more information about using SYSOPTS.

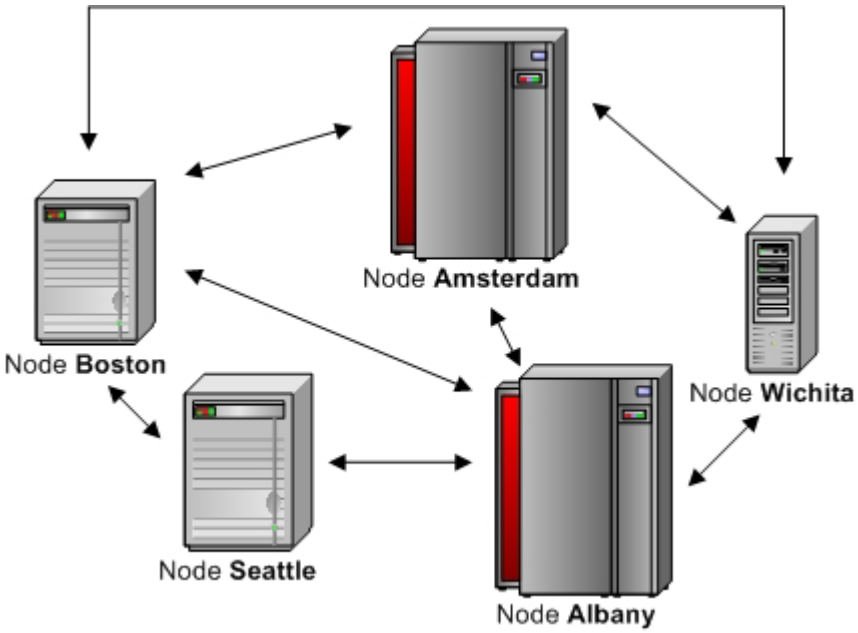
Concept of Nodes

Understanding the concept of nodes is critical to understanding how Processes work. A node is a computer in a network, as shown in the following example:



In this example, a computer in the node Boston communicates with a computer in the node Wichita.

A computer network can have numerous nodes, as shown in the following example:

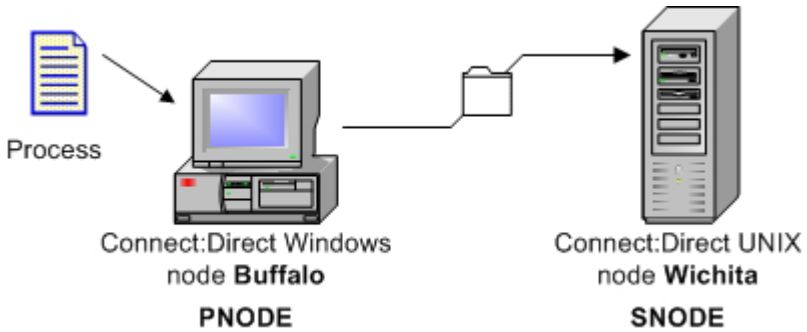


In a Sterling Connect:Direct network, each Sterling Connect:Direct server is considered a node. A Sterling Connect:Direct server can be one of two node types:

- The primary node (PNODE). This is the Sterling Connect:Direct server that initiates and controls the processing. It is the server where the Sterling Connect:Direct Process is submitted.
- The secondary node (SNODE). This is the Sterling Connect:Direct server that works with the PNODE to accomplish the processing. For example, it may be the Sterling Connect:Direct server that receives a file, or the Sterling Connect:Direct server that sends a file to the PNODE.

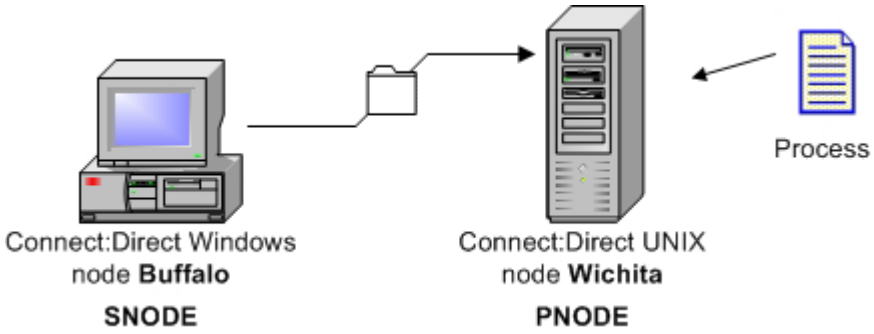
The Process defines which node is the PNODE and which is the SNODE. A node can be a PNODE in one Process and an SNODE in another Process, as shown in the following examples.

The first example shows a simple file transfer. A Process is submitted to a Sterling Connect:Direct for Microsoft Windows node to send, or push, a file to a Sterling Connect:Direct for UNIX node. In this example, the Sterling Connect:Direct for Microsoft Windows node is the PNODE and the Sterling Connect:Direct for UNIX node is the SNODE.



In the next example a Process is submitted to a Sterling Connect:Direct for UNIX node to receive, or pull, a file from the Sterling Connect:Direct for Microsoft

Windows node. In this example, the Sterling Connect:Direct for UNIX node is the PNODE and the Sterling Connect:Direct for Microsoft Windows node is the SNODE.

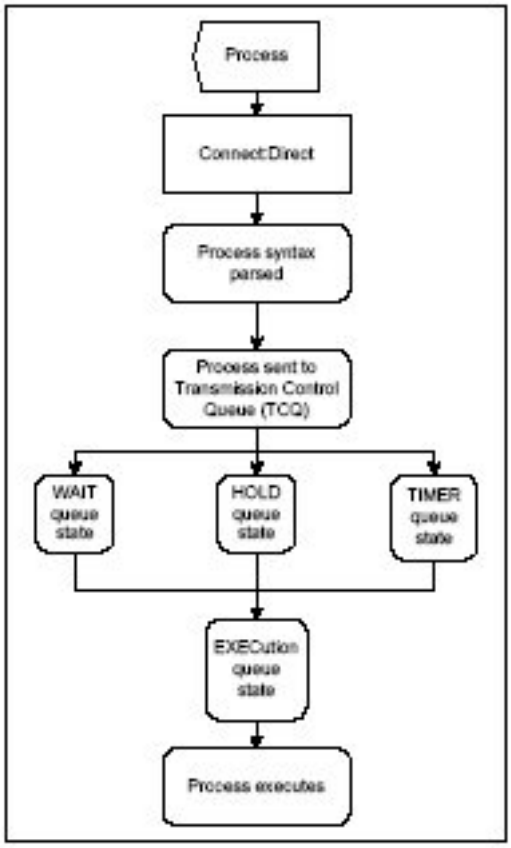


Note that the PNODE or SNODE designation has nothing to do with the file transfer direction. The PNODE and SNODE can both send or receive files.

For information about communicating with Sterling B2B Integrator, see Transmitting Files between Sterling Connect:Direct and Sterling B2B Integrator.

How Do Processes Work?

The following illustration shows how a Process executes:



The following explains the Process steps:

Step	Description
Process submitted	A user submits a Process from a Sterling Connect:Direct Process library or from the Sterling Connect:Direct Browser User Interface.
Process syntax parsed	The parser within Sterling Connect:Direct verifies the Process syntax.

Step	Description
Process sent to Transmission Control Queue (TCQ)	If the Process passes syntax checking, it is placed in the appropriate work queue according to Process parameters, such as priority, class, and start time. The Sterling Connect:Direct work queues are jointly referred to as the TCQ. A Process is in one of the following states in the TCQ: • EXECUTION–The Process is executing. • WAIT–The Process is waiting until a connection with the SNODE is established or available. Processes in the WAIT queue state may also be waiting for their turn to execute on an existing session. • HOLD–Process execution is on hold. The Process may have been submitted with a HOLD or RETAIN parameter. The Process is held on the queue until released by an operator or the SNODE connects with a request for held work. The HOLD queue state also applies to Processes that stop executing when an error occurs. • TIMER–The Process was submitted with a STARTT parameter that designates the time, date, or both that the Process should execute. Processes that initially failed due to inability to connect with the SNODE or because of a file allocation failure can also be in this queue state waiting for their retry interval to expire. Such Processes will retry automatically. A queued Process can be queried and manipulated through Sterling Connect:Direct commands such as SELECT, CHANGE, DELETE, FLUSH, and SUSPEND PROCESS. For complete information on the Sterling Connect:Direct commands and the various queues, refer to the user guide for your platform. A message indicating that the Process was submitted successfully is created when the Process is placed into the TCQ. The Process statements have been checked for syntax, but the Process may not have been selected for execution.
Process Executes	The Process is selected for execution based on Process parameters and the availability of the SNODE.

How to Build Processes

Building a Process involves analyzing the business task and creating and submitting a Process

Analyze the Business Task

The first and most important step in creating a Process is to analyze the business task you want to accomplish. Most Sterling Connect:Direct Processes copy files from one location to another (although Processes can also call external programs or even other Processes).

Consider the following factors before creating a Process.

- What files do you want to copy?
- Where do you want to copy the files from, and where to?
- What Sterling Connect:Direct platforms are involved in the transfer?
- Will the Process run at regular dates and times?
- What security is required to access the Sterling Connect:Direct servers that will execute the Process?

- Should the files be compressed during transmission?
- Do you want to perform checkpoint/restart on the transmission, enabling the transmission to resume at a defined point in case of failure?
- Who needs to be notified of Process status and completion?
- Will the Process call an external program?
- Will the Process need branching to perform additional or alternate processing, depending on previous results?

You may find it helpful to create a flow chart of the Process before creating it.

Create and Submit a Process

After you define your business need, you can create and submit a Process for execution in the following ways:

- Through the Process Builder feature of the IBM Sterling Connect:Direct Browser User Interface, a Web-based interface to a Sterling Connect:Direct server. The Sterling Connect:Direct Browser User Interface is distributed with Sterling Connect:Direct for z/OS®, HP NonStop, UNIX, and Microsoft Windows.

  The Process Builder is a Graphical User Interface that enables you to build, modify, and save Processes. The Process Builder handles Sterling Connect:Direct Process syntax rules automatically. The Process Builder eliminates the typographical mistakes made when creating Processes with a text editor. You can also validate Process syntax and submit completed Processes from the Process Builder.

  You can use the Process Builder to modify Processes created with a text editor. Likewise, Processes created with the Process Builder feature can be edited with a text editor.

  See Build Processes With the Process Builder for procedures to create a Process with the Sterling Connect:Direct Browser User Interface Process Builder.
- Through the Sterling Connect:Direct Requester for Microsoft Windows, which is a graphical interface to Sterling Connect:Direct for Microsoft Windows. See Create Processes with Requester for procedures to create a Process with the Sterling Connect:Direct Requester for Microsoft Windows.
- A text file that is submitted to a Sterling Connect:Direct server through a batch utility, command line, or a user written program through the Sterling Connect:Direct Application Program Interface (API). See Create Processes with a Text Editor for procedures to create a Process with a text editor.
- Through Sterling Connect:Direct for z/OS. See the *IBM Sterling Connect:Direct for z/OS User Guide* for information about the IUI.

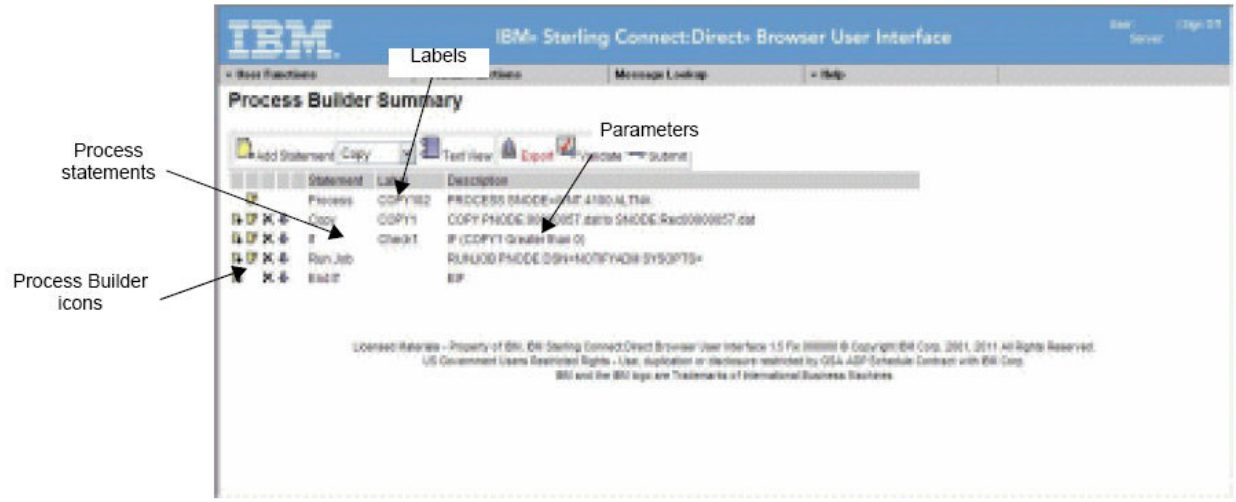
Chapter 2. Building Sterling Connect:Direct Processes With the Process Builder

Build Sterling Connect:Direct Processes With the Process Builder

The Process Builder is a graphical interface within the Sterling Connect:Direct Browser User Interface that enables you to build, modify, and save Processes. The Process Builder handles Sterling Connect:Direct Process syntax rules automatically, such as inserting quotes in SYSOPTS statements. The Process Builder eliminates the typographical mistakes made when creating Processes with a text editor. You can also validate Process syntax and submit completed Processes from the Process Builder.

Note: The maximum size allowed for a Process is 1 MB.

The following is the Process Builder home page:



The following table explains the icons on the Process Builder Summary page:

Button	Description
	Displays the Process Builder Process Statement Main Options page to create a new Process.
	Imports the specified Process into the Process Builder.
	Displays a data entry page for the statement type specified in the list box.
	Switches from a graphical view of the Process to a text view.
	Saves the Process in text format in a specified location.

Button	Description
	Sends the Process to the Sterling Connect:Direct server for Process syntax validation.
	Submits the Process to a Sterling Connect:Direct server.
	Adds a new Process statement using the parameters from the current statement.
	Edits the statement.
	Deletes the statement.
	Moves the statement down in the Process.
	Indicates that the Process statement contains invalid syntax.

Building a Process with the Process Builder consist of the following tasks:

- Signing on to the Sterling Connect:Direct Browser User Interface
- Creating a Process statement
- Adding statements for the tasks you want to accomplish
- Validating the Process
- Executing the Process

Signing on to the Sterling Connect:Direct Browser User Interface

Procedure

1. Enter the URL for the Sterling Connect:Direct Browser User Interface through your browser. Acquire this URL from the system administrator if you do not know it.
2. Select the Sterling Connect:Direct node to sign on to from the **Select Node** box, or type the node name. If this node is already configured in the Sterling Connect:Direct Browser User Interface, go to Step 5.
3. Type the IP address or host name of the Sterling Connect:Direct system that you want to sign on to.
4. Type the port number of the Sterling Connect:Direct system that you want to sign on to.

   Note: If the node you are signing on to is already configured in the Sterling Connect:Direct Browser User Interface, you can leave the three previous fields blank.
5. Specify the protocol to use. Default specifies to use the value defined in the node. If there is no node definition, default specifies to use TCP/IP.
6. Type your user ID.
7. Type your password.
8. Click **Sign On**.

Creating a Process Statement

About this task

Every Process begins with a Process statement that defines general Process information. In the Sterling Connect:Direct Browser User Interface, the Process statement consists of five pages of options that define various processing options, such as when and where the Process is submitted, if a user is notified when a task completes, who can run the Process, and symbolic variables to be substituted in the Process.

Procedure

1. From the **User Functions** menu, select **Process Builder** to display the Process Builder Summary page.
2. Click 

to display the Process Builder Process Statement Main Options page.

   Asterisks indicate required fields on this page.
3. Type a name for the Process.
4. Type the **PNODE** for the Process.

   This field already contains the node that you are signed on to. You can change it to any defined node. You do not need to be signed onto the node to specify it.
5. Select the **PNODE platform**.
6. Type the **SNODE** for the Process.
7. Select the **SNODE platform**.
8. Select another Process Statement option, or click • to return to the Process Builder Summary page.

Setting Options to Control When a Process Executes and Notification

Procedure

1. Click **Control** to display the Process Builder Process Statement Control Options page.

   All fields on this page are optional.
2. Type the **Start Time** if you want to the process to execute at a particular time.
3. Select the **Hold Status**.
4. Select the **Priority** for the Process.
5. Select the **Retain Option**.
6. Select the **class**.
7. Type the **user ID** of the person to notify when the Process finishes. This feature is not available for Sterling Connect:Direct for HP NonStop.
8. Select another Process Statement option, or click • to return to the Process Builder Summary page.

Setting Process Security Options

Procedure

1. Click **Security** to display the Process Builder Process Statement Security Options page.

All fields on this page are optional.

2. Type the **PNODE User ID**.
3. Type the **PNODE Password**.
4. Type the **SNODE User ID**.
5. Type the **SNODE password**.
6. Select another Process Statement option, or click • to return to the Process Builder Summary page.

Supplying Accounting Data

About this task

Accounting data is a free-form information that you define and use to track Process execution and data transfers. You can track data transfers by cost centers, department numbers, satellite locations, or any other type of code or identification that would benefit the management of data tracking.

Procedure

1. Click **Accounting** to display the Process Builder Process Statement Accounting Data page.

   All fields on this page are optional.
2. Type a text string to use as accounting information for the PNODE. You can enter up to 256 characters.
3. Type a text string to use as accounting information for the SNODE. You can enter up to 256 characters.
4. Select another Process Statement option, or click • to return to the Process Builder Summary page.

Defining Symbolic Variables

About this task

You use the Symbolic Variables page to specify or override symbolic variables when submitting a Process. Sterling Connect:Direct substitutes the assigned value for the variable during Process execution.

Procedure

1. Click **Variables** to display the Process Builder Process Statement Symbolic Variables page.

   All fields on this page are optional.
2. Type the variable names and values you have created.
3. Select another Process Statement option, or click • to return to the Process Builder Summary page.

Creating a Copy Statement

About this task

The Copy statement copies a file from one Sterling Connect:Direct node to another. The Copy Statement page contains several subpages that allow you to specify copy options. Copy options vary according to platform.

Procedure

1. From the Process Builder Summary page, select **Copy** and

   to display the Process Builder Copy Statement Main Options page.

   Asterisks indicate required fields on this page.
2. Type a **label** for the Copy step.
3. Select the **Copy Direction**.
4. Type the **Source File name**. If the file is on the PNODE, you can click  to browse to the file.

   Note: The browse feature is only available on Sterling Connect:Direct for z/OS 4.4, Sterling Connect:Direct for UNIX 3.6, Sterling Connect:Direct for Microsoft Windows 4.1, and Sterling Connect:Direct for HP NonStop 3.4 or later releases. Special characters (such as single quotes) in the directory name or file name are not supported for the browse feature.
5. Select the **Source DISP**.
6. Type the **Destination File name**. If the file is on the PNODE, you can click  to browse to the file.
7. Select the **Destination DISP**.
8. Select the **Compression** characteristics if you want to compress the file during transmission.
9. Select the **Checkpoint/Restart** characteristics if you want Sterling Connect:Direct to set checkpoints when it transmits the file.
10. Select another Copy Statement option, or click . to return to the Process Builder Summary page.

Specifying Copy From Options

About this task

You use the Copy Statement From page to specify additional copy options for the platform that you are copying a file from. You are presented with options for the operating system you are copying from.

The following procedure describes the Copy From options for the z/OS platform. See the Sterling Connect:Direct Browser User Interface Help for information about other platforms

Procedure

1. Type the **Source File Name**. If the file is on the PNODE, you can click  to browse to the file.
2. Type the **Data Control Block** (DCB).
3. Select the disposition of the source file using the following three subfields:
   - **Access**
   - **Normal Term**

- **Abnormal Term**

4. Type the **Unit address**.
5. Type the **Volume number**.
6. Type the **Label**.
7. Type the **SYSOPTS**.
8. Do one of the following:
   - Click to return to the Process Builder Summary page
   - Click 

to return to the Process Builder Copy Statement Main Options page.
   - Click 

to display the Process Builder Copy Statement To Options page.

Specifying Copy To Options

About this task

Use the Copy Statement To page to specify options for the platform that you are copying a file to. You are presented with options for the operating system you are copying to.

The following procedure describes the Copy To options for the z/OS platform. See the Sterling Connect:Direct Browser User Interface Help for information about other platforms

Procedure

1. Type the **Destination File Name**. If the file is on the PNODE, you can click 

to browse to the file.
2. Type the **Data Control Block (DCB)**.
3. Select the destination file disposition using the following three subfields:
   - **Access**
   - **Normal Term**
   - **Abnormal Term**
4. Type the **Space** value.
5. Type the **Unit address**.
6. Type the **Volume number**.
7. Type the **Label**.
8. Type the **Typekey**.
9. Type the **SYSOPTS**.
10. Do one of the following:
    - Click to return to the Process Builder Summary page
    - Click 

to return to the Process Builder Copy Statement Main Options page.

- Click Copy From to display the Process Builder Copy Statement From Options page.

Creating a Run Task Statement

About this task

The Run Task statement calls external programs to run on a Sterling Connect:Direct node. The Process waits until the external program completes before continuing. When the external program completes, the Run Task returns a completion code that indicates program success. This completion code can be used by subsequent Process conditional statements (see Creating Conditional Statements).

You can pass parameters to the external program as Run Task statement SYSOPTS.

The following are items to remember when writing programs called by Run Task statements:

- Do not specify programs in the Run Task statement that require user intervention.
- Do not use a completion code of 16 in the external program, or the Process will fail.

Procedure

1. From the Process Builder Summary page, select **Run Task** and 

to display the Process Builder Run Task Statement page.
2. Type a label for the Run Task step.
3. Select whether the Run Task is submitted from the PNODE or on the SNODE.
4. Type the full path to the external program. If the program is on the PNODE, you can click 

to browse to it.
5. Type the System Operations (SYSOPTS) or any optional parameters to pass to the program.
6. Click . to return to the Process Builder Summary page.

Creating a Run Job Statement

About this task

The Run Job statement executes external programs or commands to run on a Sterling Connect:Direct node. These programs run concurrently with the Process. Unlike the Run Task statement (see Creating a Run Task Statement), the Process does not wait for the program to finish.

When the Run Job statement completes, it returns a completion code. This completion code indicates the success of the Run Job statement, and not the success of the program or command.

You can pass parameters to the external program as Run Job statement SYSOPTS.

Procedure

1. From the Process Builder Summary page, select **Run Job** and 

to access the Process Builder Run Job Statement page.
2. Type a label for the Run Job step.
3. Select whether the Run Job is submitted from the PNODE or on the SNODE.
4. Type the full path to the external program. If the program is on the PNODE, you can click  to browse to it.
5. Type the System Operations (SYSOPTS) or any optional parameters to pass to the program.
6. Click • to return to the Process Builder Summary page.

Creating Conditional Statements

About this task

Conditional statements are used to branch processing within a Process, based on the result of a previous Process step. For example, when a Process performs a file copy, a conditional statement can test if the copy was successful. If the copy was successful, the Process continues processing subsequent statements. If the copy was unsuccessful, the Process can call a user-defined program that sends an error to the console operator and stops processing.

Conditional statements test against the completion code of a previous step. Conditional statements can be nested so that a Process can test for multiple results and react accordingly.

This procedure shows how to create a simple conditional statement that tests if a copy was successful. If the copy was successful, the Process executes the next statement. If the copy was not successful, the Process executes a Run Task that calls another program.

Procedure

1. Use the Process Builder to create a new Process (see Creating a Process Statement).
2. Create a Copy statement and label it `STEP1` (see Creating a Copy Statement).
3. From the Process Builder Summary page, select **If** and 

to access the Process Builder If Statement page.
4. Type a label for the If step.
5. Select the label that you want to test against from the list box (`STEP1` in this example).

   You can also type the name in the Label field, if you have not created the label yet.
6. Select **GT** (greater than) from the **Operator** field.
7. Select **4** from the **Value** field.
8. Click • to return to the Process Builder Summary page.

   The If statement is added to the Process. Note that Process Builder appends Then to the end of the statement.

9. From the Process Builder Summary page, select **Run Task** and 

to access the Process Builder Run Task Statement page.
10. Create a Run Task statement that calls another program (Creating a Run Task Statement).
11. Click • to return to the Process Builder Summary page.
12. Select **Endif** and 

.
The conditional statement is completed. You can continue adding other statements as necessary to continue processing.

Creating a Submit Statement

About this task

The Submit statement submits another Process from within a Process. The submitted Process can execute on either the PNODE or SNODE.

Procedure

1. From the Process Builder Summary page, select **Submit** and 

to display the Process Builder Submit Statement page.
2. Type a label for the Submit step.
3. Select whether the Process is submitted from the PNODE or on the SNODE.
4. Type the full path and name of the Process. If the Process is on the PNODE, you can click  to browse to it.
5. Type a new 1-8 character name for the submitted Process, if you want to rename it when it executes.
6. Click • to return to the Process Builder Summary page.

Validating Process Syntax

Procedure

To validate Process syntax, click 

from the Process Builder Summary page.
The Process is sent to the Sterling Connect:Direct server for syntax validation. If the Process syntax is valid, the message `Validation completed successfully` is displayed.
If the Process syntax is not valid, an error message is displayed. Investigate and correct any errors before submitting the Process.

Saving and Submitting a Process

Procedure

1. From the Process Builder Summary page, click 

.
   The Process Save page is displayed.
2. Click the file name **Link** to display the Save As window.

   Note: On some Web servers, clicking the file name **Link** displays the Process instead of the Save As window. If this happens, click the browser's **Back** button to return to the Process Save page. Then right-click the file name **Link** and select **Save Target As**.
3. From the Save As window, select a location to save the Process file, rename the file if desired, and click **Save**.
4. To submit a Process, click 

from the Process Builder Summary page.

Viewing a Process in Text Format

Procedure

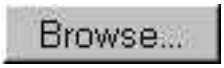
1. From the Process Builder Summary page, click 

.
   The Process is displayed in text format. This view is read-only. You cannot make changes in this view.

   Note: Sterling Connect:Direct Browser User Interface adds a /*BEGIN_REQUESTER_COMMENTS comment block to the beginning of the Process. This block contains operating system and PNODE/SNODE information. This comment block maintains compatibility with Processes built with the Sterling Connect:Direct Requester.
2. Click to return to the Process Builder Summary page.

Editing a Process

Procedure

1. From the Process Builder Summary page, click 

to navigate to and select the Process.
2. Click 

.
   The Process statements are displayed.
3. Click one of the following icons next to a Process statement to edit that statement:

Icon	Description
	Adds a new Process statement using the parameters from the current statement.
	Edits the statement.

Icon	Description
	Deletes the statement.
	Moves the statement down in the Process.

4. When finished, click 

to save the file.

Chapter 3. Creating Processes with a Text Editor

Create Processes with a Text Editor

A Sterling Connect:Direct Process is a series of statements and parameters that perform data movement and manipulation activities such as moving files, running jobs, programs, and commands, and handling processing errors.

A Sterling Connect:Direct Process uses its own scripting language that defines the work that you want the Process to do. This topic describes how to create a Process using a text editor, such as Notepad (or an ISPF Sterling Connect:Direct IUI panel on a mainframe). It provides several example Processes.

Note: The maximum size allowed for a Process is 1 MB.

Creating a Process That Executes at the Same Time Every Day

About this task

This example creates a simple Process that executes at 4:00 p.m. daily. The Process runs on a Sterling Connect:Direct for z/OS. The parameters are:

Parameter	Value
PNODE	CDZOS.NEW.YORK
SNODE	CDZOS.NEW.YORK

Procedure

1. Open a text editor (on a mainframe, create a new member in the PDS public process library dataset).
2. Type a 1-8 character name for the Process. This name must begin with an alphabetic charter in position 1. In this example, the Process is named DAILYPRC.

```
DAILYPRC
```

3. Type a few spaces, then type the keyword PROCESS.

```
DAILYPRC   PROCESS
```

   The number of spaces separating keywords does not matter. Use whatever helps you understand the statement.
4. Type a few spaces, then type the PNODE parameter followed by the PNODE value.

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK
```

5. Because we will add additional parameters for the PROCESS statement, we must use a continuation mark to show that parameters continue on a separate line. So, add a few more spaces, then type a hyphen and press RETURN to move to the next line.

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
```

6. Type some spaces until you are under the PNODE parameter, then type the SNODE parameter and its value.

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
                     SNODE =CDZOS.NEW.YORK
```

7. Type some more spaces, then type a hyphen continuation mark and press RETURN.

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
                     SNODE =CDZOS.NEW.YORK                        -
```

8. Type some spaces until you are under the SNODE parameter, then type the STARTT parameter and its value (4:00 p.m. in this example).

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
                     SNODE =CDZOS.NEW.YORK                        -
                     STARTT=(,4:00:00PM)
```

The value *(,4:00:00PM)* indicates that this Processes will run daily at 4:00 p.m.

9. Type some more spaces, then type a hyphen continuation mark and press RETURN.

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
                     SNODE =CDZOS.NEW.YORK                        -
                     STARTT=(,4:00:00PM)                        -
```

10. Type some spaces until you are under the STARTT parameter, then type RETAIN=YES

```
DAILYPRC   PROCESS   PNODE=CDZOS.NEW.YORK                          -
                     SNODE =CDZOS.NEW.YORK                        -
                     STARTT=(,4:00:00PM)                        -
                     RETAIN=YES
```

11. The RETAIN parameter saves the Process on the Process queue after it executes, so that it can run again.
12. Save the file to a library that can be accessed by Sterling Connect:Direct.
13. Use the Sterling Connect:Direct SUBMIT command to submit the Process (or use the SB option on a Sterling Connect:Direct for z/OS for IUI panel.)

Creating a Process That Copies a File

About this task

This example creates a Process that copies a file from one z/OS node to another. The copy excludes all members that begin with DM. It allocates a data set on the destination node using the source file attributes. After the copy completes, a user notification is sent. The parameters are:

Parameter	Value
PNODE	CDZOS.NEW.YORK
SNODE	CDZOS.CHICAGO

Parameter	Value
User ID to notify	NYC6
File to copy	NYC1.LINKLIB

Procedure

1. Open a text editor (on a mainframe, create a new member in the PDS public process library dataset).
2. Type a 1-8 character name for the Process. In this example, the Process is named PROCCOPY.

```
PROCCOPY
```

3. Type a few spaces, then type the PROCESS keyword and the PNODE parameter and values.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK
```

4. Type some more spaces, then type a hyphen continuation mark and press RETURN.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                    -
```

5. Type spaces until you are under the PNODE keyword, then type SNODE parameter and value, followed by a hyphen continuation mark. Press RETURN.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                    -
                       SNODE=CDZOS.CHICAGO                     -
```

6. Under the SNODE keyword type the NOTIFY parameter followed by the user ID to notify upon Process completion. As this is the last parameter for the PROCESS keyword, do not add a continuation mark.

   Press Enter.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                    -
                       SNODE=CDZOS.CHICAGO                     -
                       NOTIFY=NYC6
```

7. Next we will add the COPY statement. Type a 1-8 character label for the statement, beginning in position one. In this example, the label is called STEP1.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                    -
                       SNODE=CDZOS.CHICAGO                     -
                       NOTIFY=NYC6
STEP1
```

8. Type some spaces, then type the COPY FROM statement followed by an open parenthesis, the DSN parameter and value.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                    -
                       SNODE=CDZOS.CHICAGO                     -
                       NOTIFY=NYC6
STEP1      COPY FROM   (DSN=NYC1.LINKLIB
```

9. Type some more spaces followed by a hyphen continuation mark and press RETURN.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                  -
                       SNODE=CDZOS.CHICAGO                   -
                       NOTIFY=NYC6
STEP1      COPY FROM   (DSN=NYC1.LINKLIB                   -
```

10. Type some spaces followed by the EXCLUDE parameter.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                  -
                       SNODE=CDZOS.CHICAGO                   -
                       NOTIFY=NYC6
STEP1      COPY FROM   (DSN=NYC1.LINKLIB                   -
                       EXCLUDE=
```

11. We want to exclude all members that begin with DM from the copy, so type (DM*) followed by a close parenthesis (to close the DSN parameter) and the hyphen continuation mark.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                  -
                       SNODE=CDZOS.CHICAGO                   -
                       NOTIFY=NYC6
STEP1      COPY FROM   (DSN=NYC1.LINKLIB                   -
                       EXCLUDE=(DM*))                      -
```

12. Press RETURN to move to the next line. Type spaces until you are under FROM, then TYPE the TO parameter followed by the DSN to assign the file on the destination node.

```
PROCCOPY   PROCESS     PNODE=CDZOS.NEW.YORK                  -
                       SNODE=CDZOS.CHICAGO                   -
                       NOTIFY=NYC6
STEP1      COPY FROM   (DSN=NYC1.LINKLIB                   -
                       EXCLUDE=(DM*))                      -
                 TO    (DSN=CHI6.NEW.LINKLIB)
```

13. Save the file to a library that can be accessed by Sterling Connect:Direct.
14. Use the Sterling Connect:Direct SUBMIT command to submit the Process.

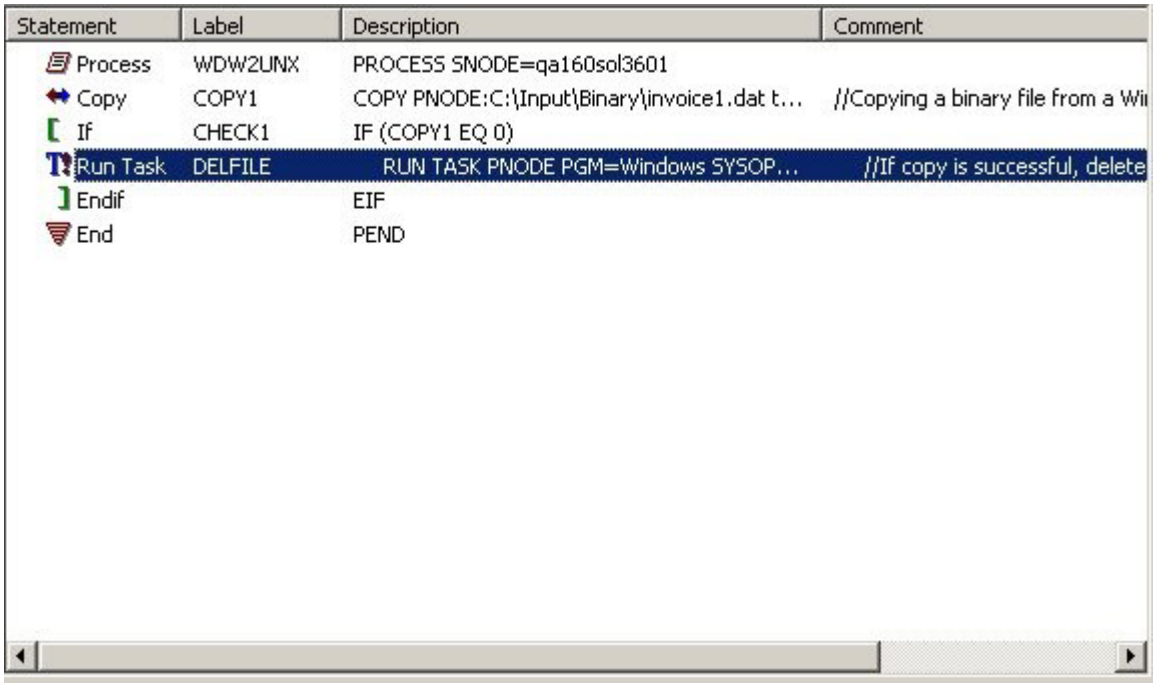
Chapter 4. Creating Processes with Sterling Connect:Direct Requester

Create Processes with Sterling Connect:Direct Requester

Sterling Connect:Direct Requester for Microsoft Windows provides a graphical interface to build, modify, and save Processes. You can select parameters from a drop-down list and automatically insert the correct syntax for each parameter. After you define a Process, you can validate the syntax.

Note: The maximum size allowed for a Process is 64 KB.

The following is a sample Process built using Sterling Connect:Direct Requester for Microsoft Windows. The Process copies a binary file from a Microsoft Windows node to a UNIX node. If the copy is successful, a run task statement is performed on the Microsoft Windows node to delete the source file on the Microsoft Windows node.



To make it easier to create a process using Sterling Connect:Direct Requester for Microsoft Windows, first attach to the node where the Process will begin. This gives you access to the network map. Since the nodes you communicate with are defined in the network map, accessing a network map allows you to identify the node that communicate with in the Process you are defining.

Building a Process in Sterling Connect:Direct Requester for Microsoft Windows requires the following tasks:

- Creating a Process Statement
- Adding other statements to perform the work
- Validating the Process

- Saving the Process file

The Sterling Connect:Direct Requester for Microsoft Windows Help contains more detailed information about how to create a Sterling Connect:Direct Process.

Creating a Process Statement

About this task

Every Process begins with a Process statement.

Procedure

1. Select **Start > Programs > Sterling Commerce Connect Direct > CD Requester**.
2. From the **File** menu, select **New** > **Process**.
3. Type a Process name, from 1 to 8 alphanumeric characters, in the **Process Name** field.
4. Type or select the name of the PNODE in the **Pnode Name** field.
5. To issue a warning message if an attempt is made to submit the Process on a different PNODE, click **Warn if submitted to a different node**.
6. To issue a warning message if an attempt is made to submit the Process on a PNODE with a different operating system, click **Warn if submitted to a different operating system**.
7. Specify the SNODE by selecting the node from the drop-down menu, typing the name of the node, or specifying an IP address and port, using the following format:

```
hostname|IPaddress;portnumber|servicename
```

8. Click one of the tabs to further define the Process options.

Identifying When to Run a Process

Procedure

1. Open the Process and click the **Control** tab.
2. Specify the date to run the Process: **Today**, select a specific **Date**, or identify a day to run the Process
3. Specify the time to run the task. Options include:
   - **Immediate** to run the Process as soon as it is activated. This option is only available if you selected **Today** or **Date** in the **Start Date** field.
   - **Time** to run the Process at a designated time. Type a time to start the Process in the format hh:mm:ss.
4. Click **OK** to save your changes and close the dialog box or click another tab to continue defining other control functions.

Placing the Process in the Hold or Retain Queue

To place the Process in the Hold or Retain queue to run at a later time:

Procedure

1. Open the Process and click the **Control** tab.
2. To place the Process in the Hold queue for future execution, select one of the following options in the **Hold** field:

- Select **Yes** to hold the Process in the queue in Held Initially status (HI) until explicitly released.
- Select **Call** to hold the Process until a connection is established between the PNODE and the SNODE. The Process executes if another Process establishes connection between the nodes.

3. To place the Process in the Retain queue, select one of the following options in the **Hold** field:
   - Select **Yes** to retain the Process in the Hold queue in Hold Retain status (HR) after execution.
   - Select **Initial** to retain the Process in the Hold queue in HR status for automatic execution every time the Process Manager initializes.
4. Click **OK** to close the dialog box or click one of the other tabs to continue modifying Process options.

   Note: If you select **Yes** as the value in the **Retain Execution Option** and you specify a start time, the value defined in the **Hold Execution Option** takes precedence. If you set the value of the **Hold Execution Option** to **Call**, and set **RETAIN** to **Yes**, the value in the **Hold Execution Option** is ignored.

Notifying a User when a Process Runs

Procedure

1. Open the Process and click the **Control** tab.
2. Type the user ID in the **Notify Userid** field.
3. Click **OK** to close the dialog box or click one of the other tabs to continue modifying Process options.

Specifying User IDs and Passwords for PNODE and SNODE

Procedure

1. Open the Process or command and click the **Security** tab.
2. Type the PNODE user ID in the **Pnode Userid** fields.
3. Type the PNODE password in the **Password field.**
4. Type the SNODE user ID in the **Snode Userid** field.
5. Type the SNODE password in the **Snode Password** fields.
6. To change the password for the user ID on the SNODE, type the new password in the **New Password** field.
7. Type the new password a second time in the **Verify New Password** field to validate the change.
8. Click **OK** to close the dialog box or click one of the other tabs to continue modifying Process options.

Adding a Copy Statement

About this task

A copy statement copies a file from one node to another.

Procedure

1. From the **Process** menu, select **Insert** > **Copy**.
2. To identify the step within the Process, type a label of up to 8 alphanumeric characters in the **Copy Statement Label** box.

3. To copy the file to the SNODE, select **Send**.
4. To copy a file from the SNODE, select **Receive**.
5. Type the name of the source file in the **Source Filename** field.
6. Type the name of the file in the **Destination Filename** field.
7. Select the method to use to save the destination file: **NEW** to create a new file, **RPL** to replace information in an existing file or to create a new file, if the file does not exist, or **MOD** to append the transferred information to an existing file.
8. Click **OK**.

   See the Sterling Connect:Direct Requester for Microsoft Windows Help for all Copy statement parameters.

Adding Conditional Statements

About this task

Conditional statements branch processing within a Process, based on the result of a previous Process step. For example, when a Process copies a file, a conditional statement can test if the copy was successful. If the copy was successful, the Process continues processing additional steps. If the copy was unsuccessful, the Process can call a user-defined program that sends an error message to the console and stops processing.

Conditional processing tests against the completion code of the previous step. Conditional steps can be nested so that a Process can test for multiple results and react accordingly.

This procedure shows how to create a simple conditional statement that tests if a copy was successful. If the copy was successful, the Process executes the next statement.

Procedure

1. From the **Process** menu, select **Insert > If** to add an If statement to a Process.
2. Type `CHECK1` as the label for this step in the **If Statement Label** field.
3. Select COPY1 as the step label to test against from the list box.
4. Select EQ as the comparison statement in the **Operator** field.
5. Select 0 in the **Value** field to check if the copy process was successfully performed.
6. Click **OK** to save the statement. The If statement is displayed in the Process window.
7. To add the End If statement, from the **Process** menu, select **Insert** > **End If**.
8. Click **OK** to save the statement. The Endif statement is displayed in the Process window.

Adding a Run Task Statement

About this task

You can run programs and commands by adding the Run Task statement to a Process. The Run Task statement executes programs on the PNODE or the SNODE. In the example, the Run Task statement deletes the source file from the Microsoft

Windows computer if the Copy step is successful. The Run Task statement must be added after the conditional If statement.

Procedure

1. From the **Node** menu, select **Run Task**.
2. Type `DELFILE` as the label for the Run Task step.
3. Type `Microsoft Windows` as the name of the program that executes the command.
4. Type `sysopts=cmd(del c:\data\out\invoice1.dat)` as the command to execute if the copy step is successful.
5. Click **OK**.

Validating Process Content

About this task

After creating a Process, validate the content. Validating Process content checks the syntax for errors or missing information. Validation does not check the content of the statements, only that they are formatted correctly. The Process validation sends messages to the Output window. A Validation Successful message means that the syntax is formatted correctly.

Procedure

1. Open the Process file.
2. From the **Process** menu, select **Validate**.
3. View the messages displayed in the Output window. If messages indicate invalid statements, edit the statements and validate the content of the Process again.

Saving a Process

Procedure

1. From the **File** menu, select **Save**.
2. Type a name for the Process, including the .CDP extension. Processes are saved in the Process directory.

Editing a Process

Procedure

1. From the Sterling Connect:Direct Requester for Microsoft Windows window, open the Process to edit.
2. Double-click the statement to edit.
3. Change the statement.
4. Click **OK**.
5. To delete a statement, highlight the statement and click **Delete**.
6. To add a new statement, right-click inside the Process and select **Insert**.

Chapter 5. Process Language Syntax

Process Language Syntax

This topic describes the syntax used in Sterling Connect:Direct Processes. If you use the Sterling Connect:Direct Browser User Interface Process Builder or the Sterling Connect:Direct Requester to create Processes, some of these syntax rules are performed automatically by the software.

Those following symbols are defined as special characters, delimiters, and operators in Sterling Connect:Direct:

-	hyphen	||	two vertical bars	&	ampersand
	(blank)	¬	(not sign)	'	(single quotation mark)
<	(less-than sign)	/	(slash)	"	(double quotation mark)
>	(greater-than sign)	\	(backslash)	[ ]	(brackets)
( )	(parentheses)	,	(comma)	{ }	(braces)
=	(equal sign)	.	(period)	*	(asterisk)

Note: The EBCDIC Hex value for the slash (/) is x'61' and x'EO' for the backslash (\) . The EBCDIC Hex value for the vertical bar (|) is X'4F.

Punctuation

This topic includes the following:

- Commas
- Continuation Marks
- Parentheses
- Asterisks

Commas

Commas have two functions:

- Separate items within a list
- Control the order of values specified as positional parameters

Use a comma to indicate omission of a positional parameter.

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

Do not use commas to separate multiple symbolics in a Process. Separate multiple symbolics with one or more spaces.

Continuation Marks

Use a hyphen as a continuation mark when a statement continues on multiple lines. Separate the hyphen from the preceding characters by at least one blank.

For Sterling Connect:Direct for HP NonStop: Both hyphen (-) and ampersand (&) are supported as continuation characters.

For Sterling Connect:Direct for UNIX, and OpenVMS: Continuation characters are not required.

Parentheses

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

Asterisks

Asterisks indicate generic specifications of parameters. With generics, you request information by specifying a single asterisk (*) or a character string plus an asterisk.

For example, the following FROM clause of a COPY statement selects all member names beginning with ACCT (the first four characters of the data set names) from the PDS named PDS.SOURCE.

```
COPY FROM (DSN=PDS.SOURCE SELECT=(ACCT*))
```

For Sterling Connect:Direct for i5/OS® Processes: Asterisks precede some subparameters and must be typed as shown in the statement format.

Comments

Comments include additional information within a Sterling Connect:Direct Process. Comments are allowed in the following formats:

- An asterisk (*) in column one, followed by the comment. You must use this format for statements processed by DMBATCH.
- Preceded by a slash-asterisk (/*) and followed by an asterisk-slash (*/). This format can be used after a continuation mark as well as at the beginning of a line.
- Preceded by a slash-asterisk (/*), continuing over multiple lines, and terminated by an asterisk-slash (*/). The terminating */ cannot begin in column one.

The following example shows every way that you can use comments.

```
/* This type of comment can be written on one line*/
/*
It can also continue across multiple lines.  Remember that
the terminating asterisk-slash cannot begin in column one.
 */
COPY      FROM (                                -             /* INPUT */
               DSN=&DSN1                        - /* SYMBOLIC DATA SET */
                UNIT=SYSDA)
* After submitting this Process,
* enable the Connect:Direct UNIX node.
```

Concatenation

Concatenation joins separate character strings into a single string. Different platforms use different characters to indicate concatenation.

Concatenation for Sterling Connect:Direct for z/OS, VM/ESA, VSE/ESA, and i5/OS

The operator blank-vertical bar-vertical bar-blank (||) indicates concatenation. For example:

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

Resolves to:

```
DSN=CDNODE
```

Concatenation for Sterling Connect:Direct for HP NonStop

Sterling Connect:Direct for HP NonStop supports two concatenation operators:

- Vertical bars (||)
- Ampersand (&) for the PACCT and SACCT parameters

Typically, concatenation is not necessary for any other parameters.

The following example PROCESS statements illustrate the differences between the two types of concatenation for the SACCT parameter. In both examples, ampersands are used to indicate continuation of the PROCESS statement.

The following example shows an ampersand (&) used as a concatenation character. When used for concatenation, the ampersand (&) must be in column 80 and the remainder of the string must begin in column one of the next line to ensure that blanks are not added to the string. The entire string must be enclosed in single quotation marks.

```
PROC1        PROCESS             SNODE=CD.OS390.NODE                                  &
           SACCT='12345678901234567890123456789012345678901234567890123456789           &
01234567890123456789'                                                                   &
           SNODEID=(USERID,PASWRD)
```

The following example shows two vertical bars (||) used as a concatenation character. Do not use blanks before or after the two vertical bars (||) to prevent blanks from being added to the string. Each line of the SACCT string must begin and end with single quotation marks.

```
PROC1       PROCESS        SNODE=CD.OS390.NODE                                    &
         SACCT='12345678901234567890'||                                          &
                '1234567890123456789'                                            &
         SNODEID=(USERID,PASWRD)
```

Concatenation for Sterling Connect:Direct for OpenVMS

Either double quotation marks (“ ”) or single quotation marks (‘ ’) can be used with the continuation character to concatenate a string spanning multiple records. The second and subsequent records must begin in column one. For example:

```
CONCAT              PROCESS             SNODE=CD.VMS.NODE
                 SYMBOL &TO="$DISK1:"                                  -
"<DIRECTORY>TEST.DAT"
STEP01              COPY FROM (DSN=IBMFILE SNODE)                      -
                      TO (DSN=&TO PNODE DISP=NEW)
```

Concatenation for Sterling Connect:Direct for UNIX and Sterling Connect:Direct for VM/ESA

Use stream input instead of concatenation or continuation symbols as required by other Sterling Connect:Direct operating environments. Also, grammar is based on the sequence of parameters and arguments instead of position within the inline buffer. The exception is that comment identifiers (asterisks [*] and pound [#] signs) in column one are positional.

Concatenation is also used in conjunction with Special Purpose Bracketing and in Symbolic Substitution to join values that are represented as symbolic parameters. See the following sections for more information.

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

Resolves to:

```
DSN=CDBOB
```

Symbolic resolution occurs before concatenation.

HFS File Name Considerations

When HFS file names contain internal Sterling Connect:Direct for z/OS keywords, single quote delimiters are required. For example, coding:

```
DSN=/u/myspace/DCB/test.file
```

would be rejected at submit time because DCB is an internal Sterling Connect:Direct for z/OS keyword that requires an equal sign.

Surrounding the entire file name within single quotes resolves this issue unless variable substitution is being used. If a Sterling Connect:Direct for z/OS keyword is being used along with variable substitution, then a more complex syntax is required. For example, coding:

```
                SYMBOL &FILE="file"
                  :
                DSN=/u/myspace/ -
                    'DCB/' -
                    test.&FILE
```

would produce:

```
DSN='/u/myspace/DCB/test.file'
```

All Sterling Connect:Direct for z/OS keywords used in DSN= or FILE= should be treated in this manner.

The following is a list of keywords that must receive special consideration if contained in an HFS file name:

ALIAS	AVGREC	BLKSIZE	BUFND	CASE	CKPT	CLASS
COMPRESS	COPY	CRC	DATACLAS	DATAEXIT	DATATYPE	DCB
DEBUG	DISP	DSN	DSNTYPE	DSORG	ESF	EXCLUDE
FOLD	FROM	HOLD	IOEXIT	JOB	KEYLEN	KEYOFF
LABEL	LIKE	LIMCT	LRECL	MGMTCLAS	MVSGP	NCP
NETMAP	NEWNAME	NODE	NOREPLACE	NOTIFY	OPTCD	PACCT
PARM	PDS.DIR	PDS.DIRECTORY	PGM	PNODE	PNODEID	PRECOMP
PRINT	REQUEUE	RESGDG	RETAIN	RKP	RUN	SACCT
SECMODEL	SELECT	SIGNON	SNODE	SPACE	SQL	STORCLAS
STRIP.BLANKS	SUB	SUBMIT	SUBNODE	SYSOPTS	TASK	TMPDD
TMPDSN	TO	TODAY	TOMORROW	TRTCH	TYPE	UNIT
USERID	VOL	VOLSER	WHERE	XLATE		

Special Character Strings

To maintain a special characters as part of a string, enclose the string within bracketing characters. The bracketing characters are:

- backslashes (\)
- single quotation marks (‘ ’)
- double quotation marks (“ ”).

Backslashes

Backslashes indicate a character string and are not maintained as part of the string at its final resolution. The following table indicates the platforms that accept backslashes:

Processes Submitted from	**Backslashes valid?**
HP NonStop	No
OpenVMS	No
z/OS	Yes
i5/OS	Yes
UNIX	No
VM/ESA	Yes
VSE/ESA	Yes
Microsoft Windows	No

Use backslashes to:

- Continue a string containing special characters across multiple lines.
- Ensure that quotation marks in the string are maintained.

Both backslashes must be on the same line. If a string containing special characters continues across multiple lines, each line containing a special character must be enclosed in backslashes and concatenated. For example, the following SYSOPTS parameter for Sterling Connect:Direct for i5/OS is a quoted string and must be enclosed in backslashes when it continues across multiple lines:

```
SYSOPTS=    \"CMD(\          ||                -
            \SNDBRKMSG\      ||                -
            \)"\
```

Resolves to:

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

If the character string includes a backslash, precede it with an additional backslash. For example:

```
PACCT= \xd4 DEPT\MIS\        ||       -
      \602'\
```

Resolves to:

```
PACCT='DEPT\MIS602'
```

Single and Double Quotation Marks

Use single and double quotation marks to embed special characters or blanks within a parameter or subparameter value. For example:

```
COPY TO (DSN='VMFILE FILETYPE')
COPY TO (DSN=\"C:\PCDIR\BAT.EXE"\)
```

Strings within apostrophes (single quotes) allow the parsing of parameters as entered. Strings within quotes (double quotes) allow the resolution of &values in a quoted string.

For Sterling Connect:Direct for i5/OS and OpenVMS: Enclose the entire SYSOPTS string in double quotation marks (“ ”).

For Sterling Connect:Direct for z/OS: Enclose parameters in double quotation marks (“ ”).

For Sterling Connect:Direct for UNIX: Use double quotation marks, unless otherwise specified.

Special Character Parsing in the SUBMIT Command

Parsing of special bracketing and single and double quotes is performed differently in a SUBMIT command than in a SUBMIT statement within a Process.

For example, a SUBMIT command executed from DMBATCH, resolves:

```
SYMBOL  &BATCHID2=\'''\ || BATCHID || \'''\
```

to:

```
'''BATCHID'''
```

A SUBMIT statement within a Process resolves the same string to:

```
'BATCHID'
```

This is important to remember when a SUBMIT is performed between different platforms and products, such as from Sterling Connect:Direct for UNIX to Sterling Connect:Direct for z/OS to IBM Sterling Connect:Enterprise®.

Symbolic Substitution

Symbolic substitution substitutes information in a Process. When Sterling Connect:Direct encounters an ampersand (&) followed by 1-8 alphanumeric characters, it substitutes a string represented by the ampersand and alphanumeric characters. For example:

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

Resolves to:

```
DSN=CDBOB
```

Separate multiple symbolics with spaces, as shown in the following:

```
SUBMIT PROC=TSTSEND &DSN1=TSTSEND.VAR0001.S200010 &RUNDATE=200012 &TSTDATE=200010
```

Symbolic resolution occurs before concatenation.

The following example encloses a string in double quotation marks to resolve the symbolic &FILTYP.

```
PROC2 PROCESS SNODE=CD.VM &FILTYP=FT
 COPY  FROM (DSN=OS390.DATA -
  DISP=SHR) -
       TO   (DSN="FN || &FILTYP" -
  LINK=(IVVB,WIVVB,W,191) -
  DISP=(RPL))
```

Double quotation marks are not valid for symbolic substitution in Microsoft Windows. Use the SYMBOL statement and concatenation instead.

If you are using symbolic substitution for SNODEID and want to pass both a user ID and password, you must have separate symbolic names for each entity, that is, a symbolic name for the user ID and another for the password.

For example, if you code the following:

```
&MYSTUFF=myid,mypass
SNODED=(&MYSTUFF)
```

Sterling Connect:Direct will interpret &MYSTUFF as one block rather than as two entities–a user ID and password.

To pass two pieces of information in this example, code it like the following:

```
&MYNAME=myid
&MYPASS=mypass
SNODEID=(&MYNAME,&MYPASS)
```

Intrinsic Symbolic Variables Used in Sterling Connect:Direct for z/OS and Microsoft Windows

Sterling Connect:Direct provides the following intrinsic symbolic variables that you can use to substitute user-defined values when a Process is executed. This flexibility lets you use the same Process for multiple applications when these values change.

Value	Description
%DD2DSN	Specifies an allocated DD statement, which references a DSN to be passed to a Process being submitted (for Sterling Connect:Direct for z/OS)
%JDATE	Specifies the date the Process was submitted in Julian format. The variable is resolved as the submission date of the Process in the format yyyyddd. Among other uses, the value returned is suitable for constructing a file name on the node receiving the file. The value of the variable is resolved at Process submit time. The value will correspond to the date on which the Process was submitted, regardless of when or how many times the Process is actually executed.
%JOBID	Specifies the job number.
%JOBNM	Specifies the job name.
%JUSER	Specifies a variable that resolves to the USERID of the submitted job.
%NUM1	Specifies the submission time of the Process in minutes, seconds, and fraction of seconds in the format mmssth.
%NUM2	Specifies the submitted time of a Process as the low order 4 bits of the milliseconds of the time expressed as 1 hex digit (a value from 0 through 15 expressed as 0 through F).
%PNODE	PNODE name where the submit occurs
%PRAND	Pseudo-random number (6 hex digits)
%SUBDATE	Specifies the date the Process was submitted in Gregorian format. The variable is resolved as the submission date of the Process in the format cyymmdd where c is the century indicator and is set to 0 for year 19yy or 1 for year 20yy. The value returned can be used to create a file name on the node receiving the file.
%SUBDATE1	Use this parameter to substitute the submitted date in the yyyymmdd date format.
%SUBDATE2	Use this parameter to substitute the submitted date in the yyyyddmm date format.

Value	Description
%SUBDATE3	Use this parameter to substitute the submitted date in the mmddyyyy date format.
%SUBDATE4	Use this parameter to substitute the submitted date in the ddmmyyyy date format.
%SUBTIME	Specifies the time the process was submitted. The variable is resolved as the submission time of the process in the format hhmmss. The return value can be used to create a file name on the node receiving the file. The value of the variable is resolved at Process submit time. The value will correspond to the time at which the Process was submitted, regardless of when or how many times the Process is actually executed.
%USER	Specifies a variable that resolves to the user submitting the Process

In the following example for Sterling Connect:Direct for z/OS, the DSN specified in the FROMDD statement is DALLAS.DATA.FILE.

```
&DSN = %DD2DSN(FROMDD)
```

The DSN resolves to DALLAS.DATA.FILE when Sterling Connect:Direct for z/OS executes the Process containing the intrinsic symbolic variable

Termination

A statement is terminated by the end of data without a continuation mark.

SYSOPTS Syntax

SYSOPTS (system operations) are a specialized type of parameter used by every Sterling Connect:Direct platform. SYSOPTS specify platform-specific commands to perform during a Process. For example, when transferring a file from a mainframe system to a Microsoft Windows system, you use SYSOPTS to specify that the file be translated from EBCDIC to ASCII and that any trailing blanks be removed.

All Sterling Connect:Direct platforms use SYSOPTS on the Copy statement. Some platforms also use SYSOPTS on the Run Job and Run Task statements to pass parameters to the external program.

Because of operating system differences, SYSOPTS parameters and syntax vary by platform. This can be confusing when you create Processes with a text editor. (However, the Sterling Connect:Direct Requester and the Sterling Connect:Direct Browser User Interface Process Builder both automatically handle Process syntax.)

You must use the SYSOPTS syntax for the platform that is performing the work. For example, if you are copying a file from z/OS to HP NonStop, then performing a Run Task on the copied file, you use the Run Task's SYSOPTS syntax for HP NonStop.

If you are sending files to Sterling B2B Integrator and want to use SYSOPTS parameters to customize the format of those files, see Specifying File Formats using SYSOPTS.

Note: Depending on how you use a variable string, you may need to include bracketing characters. This situation is often required when a SYSOPTS string is sent as a symbolic parameter and must be enclosed in quotation marks.

For example, to transfer a file to a UNIX system using a symbolic variable, you would type the SYSOPTS clause as follows:

```
&SYSOPTS=\":datatype=text:xlate=yes:"\
```

In this example, what the Process states:

```
SYSOPTS=&SYSOPTS
```

resolves to:

```
SYSOPTS=":datatype=text:xlate=yes:"
```

The following list explains the SYSOPTS syntax differences between platforms. Remember, Sterling Connect:Direct Requester and the Sterling Connect:Direct Browser User Interface Process Builder handle Process syntax automatically, so you do not need to worry about these conventions if you use those tools.

Processes Submitted from HP NonStop

Copy Statement: Copy statement SYSOPTS are expressed as HP NonStop SET commands. Enclose each SYSOPTS string in double quotation marks except when copying from Microsoft Windows to HP NonStop. For example:

```
SYSOPTS=("SET parameter")
```

When copying from Microsoft Windows to HP NonStop, enclose each SET parameter in single quotation marks and enclose the entire SYSOPTS string in double quotation marks. For example:

```
SYSOPTS="'SET parameter' 'SET parameter' 'SET parameter'"
```

There are two ways to express multiple SET command parameters:

- SET precedes each parameter. For example:

```
SYSOPTS=("SET parameter" "SET parameter")
```

- SET precedes the first parameter, and commas separate subsequent parameters. For example:

```
SYSOPTS=("SET parameter, parameter")
```

Do not use continuation marks.

Run Task Statement: Enclose a Run Task statement's SYSOPTS string in either single or double quotation marks. Enclose any literal parameter values to be passed in single quotation marks. Enclose any symbolic values (&value) in double quotation marks

Processes Submitted from z/OS

Enclose the complete SYSOPTS string in double quotation marks. Separate individual SYSOPTS parameters with spaces and use the delimiter appropriate to the platform. For example, if you are copying to Microsoft Windows, you would enclose keyword values in parentheses:

```
SYSOPTS = "DATATYPE(BINARY) XLATE(YES) STRIP.BLANKS(NO)"
```

Use backslashes (\) and concatenation characters (||) to continue the SYSOPTS string over multiple lines when the Process. For example:

```
SYSOPTS=\"TYPE(MBR)\
        \TEXT('CREATED BY PROC#001')\
         \RCDLEN(133)"\
```

Processes Submitted from i5/OS

Copy Statement: Enclose all SYSOPTS parameter values in parentheses. Enclose the entire SYSOPTS string in double quotation marks. Separate subparameters with blanks. For example:

```
SYSOPTS="TYPE(FILE) PRECMPR(*YES) XTRAN(EBCXKSC) XTRANLDATA(MIXED)"
```

Run Job Statement: Enclose the string to be passed in double quotation marks. For example:

```
SYSOPTS = "string"
```

Run Task Statement: Enclose the CL command in parentheses. Enclose the entire SYSOPTS string in double quotation marks. For example:

```
SYSOPTS = "cmd(CL command)"
```

Processes Submitted from OpenVMS

Copy Statement: Enclose the SYSOPTS string in double quotation marks. Enclose each subparameter string in single quotation marks. Separate the subparameters by blanks. For example:

```
SYSOPTS="MOUNT='MUA0 TAPELABEL' NODISMOUNT"
```

Run Job Statement: Enclose the SYSOPTS string in double quotation marks. Enclose each subparameter string in single quotation marks. Separate the subparameters by blanks. For example:

```
SYSOPTS ="KEEP LOG='log-name' NOPRINT"
```

Run Task Statement: Enclose the entire SYSOPTS string in double quotation marks. Enclose each subparameter string in single quotation marks. Separate the subparameters by blanks. For example:

```
SYSOPTS="[OUTPUT='file specification']
```

If you are calling a DCL command procedure that contains embedded blanks and quotation marks, replace the embedded blanks with underscores and remove the quotation marks. For example, specify the DCL command *MAIL/SUBJECT "two words" filename* as:

```
SYSOPTS="CMD='MAIL/SUBJECT=two_words filename'"
```

Processes Submitted from UNIX

Copy Statement: Enclose the SYSOPTS string in double quotes. For example:

```
sysopts=":datatype=text:xlate=no:pipe=yes:"
```

Run Job and Run Task Statements: Enclose the SYSOPTS string in double quotes. Separate the UNIX commands with semicolons. For example:

```
sysopts = "unix command;unix command;unix command"
```

Processes Submitted from VM

Enclose the SYSOPTS string in single or double quotes. For example:

```
SYSOPTS='!SPOOL CLASS B DIST VM1500'
```

or

```
SYSOPTS="!SPOOL CLASS B DIST VM1500"
```

Processes Submitted from VSE

Enclose each SYSOPTS parameter string in double quotation marks. For example:

```
SYSOPTS = "DBCS=(tablename,so,si,PAD)" "parameter1,parameter2"
```

Processes Submitted from Microsoft Windows

Enclose the entire string in double quotation marks. Separate the parameters by spaces. For example:

```
"xlate(yes) xlate.tbl(tbl)"
```

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.

1623-14, Shimotsuruma, Yamato-shi

Kanagawa 242-8502 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 2012. Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2012.

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.

Printed in USA