Sterling Selling and Fulfillment Foundation

Customizing the Swing Interface

Version 9.1

Sterling Selling and Fulfillment Foundation

Customizing the Swing Interface

Version 9.1

Note
Before using this information and the product it supports, read the information in "Notices" on page 13.

Contents

Chapter 1. Checklist for Customization Projects

Customization Projects

Projects to customize or extend Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales vary with the type of changes that are needed. However, most projects involve an interconnected series of changes that are best carried out in a particular order. The checklist identifies the most common order of customization tasks and indicates which guide in the documentation set provides details about each stage.

Prepare Your Development Environment

Set up a development environment that mirrors your production environment, including whether you deploy your application on a WebLogic, WebSphere®, or JBoss application server. Doing so ensures that you can test your extensions in a real-time environment.

You install and deploy your application in your development environment following the same steps that you used to install and deploy it in your production environment. Refer to your system requirements and installation documentation for details.

You have an option to customize your application with Microsoft COM+. Using Microsoft COM+ has advantages such as increased security, better performance, increased manageability of server applications, and support for clients of mixed environments. If this is your choice, see the *Customization Basics Guide* about additional installation instructions.

Plan Your Customizations

Are you adding a new menu entry? Or customizing the sign-in screen or logo? Or customizing views or wizards? Or creating new themes or new screens? Each type of customization varies in scope and complexity.

For background, see the *Customization Basics Guide*, which summarizes the types of changes that you can make and provides important guidelines about file names, keywords, and other general conventions.

Extend the Database

For many customization projects, the first task is to extend the database so that it supports the other UI or API changes that you make later. For instructions, see the *Extending the Database Guide*, which includes information about the following topics:

- Important guidelines about what you can and cannot change in the database.
- Information about modifying APIs. If you modify database tables so that any APIs are impacted, you must extend the templates of those APIs or you cannot store or retrieve data from the database. This step is required if table modifications impact an API.
- How to generate audit references so that you improve record management by tracking records at the entity level. This step is optional.

Make Other Changes to APIs

Your application can call or invoke standard APIs or custom APIs. For background about APIs and the services architecture of service types, behavior, and security, see the *Customizing APIs Guide*. This guide includes information about the following types of changes:

- Invoke standard APIs for displaying data in the UI and for saving changes made in the UI to the database.
- Invoke customized APIs for executing your custom logic in the extended service definitions and pipeline configurations.
- APIs use input and output XML to store and retrieve data from the database. If you don't extend these API input and output XML files, you may not get the results you want in the UI when your business logic is executing.
- Every API input and output XML file has a DTD and XSD associated to it. Whenever you modify input and output XML, you must generate the corresponding DTD and XSD to ensure data integrity. If you don't generate the DTD and XSD for extended XMLs, you may get inconsistent data.

Customize the UI

IBM® applications support several UI frameworks. Depending on your application and the customizations you want to make, you may work in only one or in several of these frameworks. Each framework has its own process for customizing components such as menu items, logos, themes, and so on.

Depending on the framework you want, consult one of the following guides:

- *Customizing the Console JSP Interface Guide*
- *Customizing the Swing Interface Guide*
- *Customizing User Interfaces for Mobile Devices Guide*
- *Customizing the Rich Client Platform Guide* and *Using the RCP Extensibility Tool Guide*
- *Customizing the Web UI Framework Guide*

Depending on the framework you want, consult one of the following guides:

- *Customizing the Console JSP Interface Guide*
- *Customizing the Swing Interface Guide*
- *Customizing User Interfaces for Mobile Devices Guide*
- *Customizing the Rich Client Platform Guide* and *Using the RCP Extensibility Tool Guide*
- *Customizing the Web UI Framework Guide*

Extend Transactions

You can extend and enhance the standard functionality of your application by extending the Condition Builder and by integrating with external systems. For background about transaction types, security, dynamic variables, and extending the Condition Builder, see the *Extending Transactions Guide* and *Extending the Condition Builder Guide*. These guides includes information about the following types of changes:

- Extend the Condition Builder to define complex and dynamic conditions for executing your custom business logic and using a static set of attributes.

- Define variables to dynamically configure properties belonging to actions, agents, and services configurations.
- Set up transactional data security for controlling who has access to what data, how much they can see, and what they can do with it.
- Create custom time-triggered transactions. You can invoke and schedule custom time-triggered transactions in much the same manner as you invoke and schedule the time-triggered transactions supplied by your application.
- Coordinate your custom, time-triggered transactions with external transactions and run them either by raising an event, calling a user exit, or invoking a custom API or service.

Build and Deploy your Customizations or Extensions

After performing the customizations that you want, you must build and deploy your customizations or extensions.

1. Build and deploy your customizations or extensions in the test environment so you can verify them.
2. When you are ready, repeat the same process to build and deploy your customizations and extensions in your production environment.

For instructions about this process, see the *Customization Basics Guide* which includes information about the following topics:

- Building and deploying standard resources, database extensions, and other extensions (such as templates, user exits, and Java™ interfaces).
- Building and deploying enterprise-level extensions.

Chapter 2. Swing User Interface Extensibility

About Extending the Swing User Interface

The Presentation Framework allows you to customize how information is displayed without changing the way it functions. You can customize the user interface for the Applications ManagerConfigurator in several ways. Each customization is accomplished through a combination of configuration changes made in the Applications ManagerConfigurator and changes to Java Swing code.

After you are satisfied with all of the customizations you have made to the Applications ManagerConfigurator UI, create and deploy the extended UI-specific JAR files.

The main purpose of user interface extensibility is to enable any database extensions to be integrated into the graphical user interface.

Extensibility includes the following modifications:

- Adding any icons (or buttons) and labels
- Adding any text fields and checkboxes
- Hiding any non-mandatory components
- Reorganizing the components that are displayed on-screen

Note: If you extend the Swing user interface, when you install upgrades and services packs you need to read the upgrade documentation and carefully reconcile changes that have taken place in the default screens of Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales. Some changes may be mandatory while some may not be.

You may modify the following types of screens:

- Search screens
- Detail screens
- List screens

The explorer screens (screens that contain only a tree) are not extensible.

Extending Organization and Item Detail Screens

When you extend a table used by the Item Details screen or the Organization Details screen in the Applications ManagerConfigurator, this adds a pop-up action icon to the main screen. This icon enables access to a pop-up screen that contains all of the relevant extended fields.

For example, when you extend the organization table, a pop-up access icon is added to the default organization screen. When the user clicks this icon, a pop-up screen displays all of the extended attributes that are relevant to this table.

Typically, extended attributes on the pop-up screen appear as text input boxes. In the case of the Item Detail screen, any item attribute that is specified as a classification displays as a classification lookup. This lookup enables the user to

select any classification value that has been configured as described in the Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales: *Catalog Management Configuration Guide*.

Fields on the pop-up screen can be grouped together. These groups are displayed in alphabetical order on the screen. Likewise, the fields within each group are displayed in alphabetical order.

Add Extended Attributes to a Pop-Up

About this task

To add extended attributes to a pop-up:

Procedure

1. Copy the *install_dir*/repository/entity/extensions/Extensions.xml.sample file as *install_dir*/extensions/global/entities/*your_filename*.xml file OR modify your existing extension XML file.
2. Edit the *your_filename*.xml file and specify a DataType attribute for your new column. The new attribute should be a new data type that has not been defined previously in the datatypes.xml file.
3. Edit your datatypes.xml file in *install_dir*/repository/datatypes directory and add an entry for your new data type using the following attributes:
   - DisplayInUI - Required. Specifies whether the field should automatically display in the new pop-up screen.
   - DisplayGroup - Optional. Groups the display of extended fields on the new pop-up screen. When this field is specified, all extended fields with the same DisplayGroup attribute are grouped together and displayed in alphabetical order beneath a title that is set to the value of the DisplayGroup attribute.

   For example, attributes for the newly added ExtnMyField data type are specified as follows:

```
<DataType Name='ExtnMyField' Type='CHAR' Size='35'>
       <UIType Size="30" UITableSize="30" DisplayInUI="true"
DisplayGroup="My_Group"/>
</DataType>
```

Extending Search and Detail Screens

The Applications ManagerConfigurator menu structure contains a hierarchical collection of the following types of items:

- Menu – Contains child menu items
- Resource – Contains no child menu items, and instead, points to a resource

The menu structure cannot be modified.

Note: You need Net Beans 3.2 IDE to extend search and detail screens in the application.

Each screen is associated with a Form Class and a Java Behavior Class. The Form Class is responsible for painting the controls on the screen and the Behavior Class is responsible for populating data in the screen and responding to events that occur in the screen, such as choosing Save.

When navigating to a screen within the Applications ManagerConfigurator, the Form Class of the corresponding screen is loaded and the Behavior Class populates the data in the screen.

Screens are defined in an XML file. This file contains the unique screen ID, Form Class, and Behavior Class for each screen. This file must be extended in order to extract the screens.

Extend a Search or Detail Screen

About this task

To extend a search or detail screen in the Applications ManagerConfigurator:

Procedure

1. From the Applications ManagerConfigurator, navigate to the screen that you want to extend.
2. After the screen loads, press **CTRL-M**, which displays the window with the Form Name (which is the resource ID for the screen), the Form Class name, and XML data information.
3. Note the Form Name (resource ID) and Form Class Name.
4. The *install_dir*/xapidocs/code_examples/configuisrc/scfoundation*application*uisrc.jar file contains the source code corresponding to all Form classes. There is a corresponding .form file and a .java file. The .form file is used by NetBeans and is required only if you use the NetBeans 3.2 IDE.
5. Copy the .java and .form files corresponding to the Form Class Name that you had noted into your own directory structure. The copy you make should have a different class name. Make sure you do not copy it anywhere under com.yantra because that is reserved strictly for products of Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales.
6. Add the following JAR files to the CLASSPATH. This can be done in NetBeans 3.2 by mounting the JAR file.
   - jgo.jar
   - platui.jar
   - ycmui.jar
   - yifui.jar
   - xercesImpl.jar

   You need to do this in order to compile the Java file.
7. Add the package name to the top of the form. Put the appropriate class name in the code (should be the same Java class that you originally copied as the file name created in step 5).

   The copied Java class must extend the original.
8. Set the Variables Modifier option in Net Beans to `public`. The default value is `private`. This option can usually be found in **Tools** > **Options** > **Form Objects** > **Expert** Tab.
9. Remove `super.init()` from the init() function.
10. At the end of the init() function, add the following line:

    ```
    checkVars();
    ```
11. Make the necessary changes to the new form. To set the properties of the new controls, see the XML Binding table in "XML Binding in the Swing User Interface" on page 8. Only the following changes are permitted:
    - Rearranging any components on the user interface

- Hiding any non-mandatory components
- Adding any buttons and labels
- Adding any text fields and checkboxes

12. Compile the .java file, create a JAR file named yfsextn.jar that contains only the .class file, and put it in the *install_dir*/extensions/global/webpages/yfscommon/ directory.
13. Enter the name of the newly added yfsextn.jar file in the *install_dir*/repository/eardata/platform/war/yfscommon/ jarlist.txt.
14. Rename the *install_dir*/repository/xapi/template/merged/configresource/extn_application.xml.sample file to "extn_application.xml".
15. Edit the extn_application.xml file to include the Form Name (the resource ID that you noted in step 3) and the Override Form Class Name (the complete path of the new class name that overrides the existing application class).

XML Binding in the Swing User Interface

All the forms in the Swing user interface follow the Model-View-Controller (MVC) paradigm. The form itself acts as the View, so it only has presentation logic. All the business logic is in a separate Controller class. The model for every form is an XML-DOM Element.

To further simplify the presentation logic, the core classes in <yfs> support a form of XMLBinding of different types of controls to the DOM model. This enables the form designer to bind the different controls on a form to different parts of the DOM. At run-time the infrastructure keeps the DOM and the controls synchronized. When the DOM is changed the changes are reflected on the control, and the reverse is true as well.

The following controls can be bound:

- Javax.swing.JTextComponent (and any subclasses)
- Javax.swing.JTable

The binding semantics control are described in detail below.

Binding Common to All Controls

Given an input XML, parts of the XML can be bound to controls based on an XMLBindingString. Each string is evaluated in XSL syntax and the first match is used as the value of the binding.

Assume the bound XML is the following:

```
<Order OrderNo='23' OrderDate='20010101' >
     <ShipToAddress City='Nashua' />
</Order>
```

The following table illustrates the XMLBinding.

XML Binding String	Example Value
Xml:/Order/@OrderNo	23
Xml:/Order/@OrderDate	20010101
Xml:/Order/ShipToAddress/@City	Nashua
Xml:/Order/ShipToAddress	Evaluates to the ShipToAddress Node

Name Property

Every Swing control has a bean property name that can be set by using the `setName(String)` function. The value of this property should be an XMLBindingString that specifies the XMLData that is to be bound to that control.

Binding for JText Field

In addition to the Name property, the following JTextField properties can be set for any text field:

Property	Syntax

Data Type
 	
Only Integer, Date and String are supported
	`txtField1.putClientProperty("YFCXMLBinding.dataType", "Integer")`
	`txtField1.putClientProperty("YFCXMLBinding.dataType", "Date")`
	`txtField1.putClientProperty("YFCXMLBinding.dataType", "String")`

Associated Label This is the JLabel associated with the text field	`txtField1.putClientProperty("YFCXMLBinding.associatedLabel", "lblField1")`
Mandatory If Mandatory property is set to "true" and the field is left blank, the label associated with the text field changes color. This is triggered on the lostfocus event of the text field.	`txtField1.putClientProperty("YFCXMLBinding.isMandatory", "true")`

Extending List Screens

About this task

To add, remove, and rearrange columns in the list screen within the Applications ManagerConfigurator, complete the following steps.

To extend list screens:

Procedure

1. Copy *install_dir*/repository/xapi/template/merged/configresource/genericscreens_modifications.xml.sample to *install_dir*/extensions/global/template/configresource/genericscreens_modifications.xml.

   The following is an example of the file format.

```
<ScreenModifications>
  <resourceId of the list form you want to extend>
     <ListInfo>
         <Absolute>
           <List AttributeName="" ColumnTitle="" DataType=""/>
           <List AttributeName="" ColumnTitle="" DataType=""/>
        </Absolute>
         <Add>
           <List AttributeName="" ColumnTitle="" DataType=""/>
        </Add>
         <Remove>
```

```
            <List AttributeName="" ColumnTitle="" DataType=""/>
          </Remove>
        </ListInfo>
    </resourceId of the list form you want to extend>
</ScreenModifications>
```

2. Edit the genericscreens_modifications.xml file and insert entries for the resource IDs that correspond with the list screens that you want to extend.

   Tip: In order to determine the Resource ID for a specific list screen, complete the following steps:

   a. From within the Applications ManagerConfigurator, select the User Group screen.
   b. Click the Permission tab to display a hierarchical view of all permissions.
   c. Search the permission tree to find the list screen that you want to extend.
   d. Hover your mouse over the node to display the Resource ID using the tool tip.

3. Enter the ListInfo element attributes in the list modification XML file, using the descriptions listed in the following table:

ListInfo Element	Purpose
Absolute	Replaces the current columns of a list with the new columns you specify in the List elements. Overrides Add and Remove.
Add	Adds one or more columns to the list screen. Add and Remove are mutually exclusive with Absolute.
Remove	Removes one or more columns from the list screen. Add and Remove are mutually exclusive with Absolute.
List	The List element is a sub-element of the Absolute, Add, or Remove element. Adds the following attributes (which must be specified): • AttributeName - The name to display in the list attribute. • ColumnTitle - The title to display in the list in the attribute. • DataType - The data type to display in the new column. Must be a valid DataType Name within the datatypes.xml file.

4. Open the *install_dir*/repository/datatypes/datatypes.xml file to determine which data type you should specify in the DataType attribute.
5. If the data type you need is not in the datatypes.xml file, create a new data type, filling in the element attributes using the descriptions listed in the following table:

DataType Element	Purpose
Name	Identifier of the DataType. Must exactly match the List element DataType attribute used in the genericscreens_modifications.xml file.
Type	Adds the following attributes (of which, Type and Size must be specified): • CHAR - Attribute Size • DATE - Attribute Size • NUMBER - Attributes Size, DecimalDigits (optional), NegativeAllowed (optional), ZeroAllowed (optional) • VARCHAR2 - Attribute Size

6. Save your changes to the genericscreens_modifications.xml file and close it.
7. Select the refresh cache icon that fits your needs as follows:
   - If you want to update one entity and its child resources - Select the specific entity and select the  Refresh Entity Cache icon
   - If you want to update all resources - Select the  Refresh Cache icon
8. Run the application again to test your changes.

Creating and Modifying User Themes

User themes determine the set of colors used in the Applications ManagerConfigurator for graphical user interface elements such as screens, labels, and table headers.

Creating and Modifying Custom Error Codes

In the Applications ManagerConfigurator, you can create error codes to be thrown for any exception specified in your custom code. The associated cause, action and description defined while creating the error code should be available in your user interface. Custom error code values should not contain any keywords reserved by the application.

Custom error codes are also used for failure reasons thrown by the password validation user exits. These user exit exceptions are thrown when a user attempts to save the password changes in the console. A sample implementation of these user exits using Java's built-in MD5 routines are provided in *install_dir*/xapidocs/code_examples/pwcrypt directory. For more information on these user exits, refer to the Javadocs.

For more information on creating or modifying custom error codes, refer to the presentation component in the Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales: *Configuration Guide*.

Customizing Symbols for Node Types

About this task

In the Fulfillment Network Model in the Applications ManagerConfigurator, node types are displayed as symbols of various shapes, colors and sizes. Using the provided extn_mapmanager.xml.sample file you can modify the look and feel of these node types on the Fulfillment Network Model. For more information on the Fulfillment Network Model, refer to the Sterling Business CenterSterling Selling and Fulfillment FoundationSterling Field Sales: *Configuration Guide Application Platform Configuration Guide*.

To customize node type symbols:

Procedure

1. Rename the *install_dir*/repository/xapi/template/merged/configresource/extn_mapmanager.xml.sample file to "extn_mapmanager.xml".
2. Edit the extn_mapmanager.xml file to include the node type you are customizing and include the applicable information.

   Valid values for **ShapeType** are:

   - Ellipse

- Rectangle
- RoundRectangle
- Diamond
- TriangleUp
- TriangleDown
- TriangleLeft
- TriangleRight
- Marker

For **Color** and **Selected Color**, specify any hex code or standard color name.

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.

Product Number: xxxx-xxx

Printed in USA